Universidad Carlos III de Madrid

Escuela Politécnica Superior

Grado en Ingeniería Telemática

Herramienta para clasificación de Tweets en inglés

Autor: Tello Ismael Miñana Rontomé
Tutor: Luis Sánchez Fernández

Agradecimientos

En primer lugar, agradecer a mi madre el haber confiado en mí en todo momento y haberme dado la oportunidad de estudiar lo que siempre me ha gustado y proporcionarme todos los medios necesarios para lograrlo.

A mi familia, que influyeron en mi madurez para lograr todos los objetivos de mi vida. La familia es lo más importante.

También quiero agradecer al tutor de mi Trabajo Fin de Grado, Luis Sánchez, por su confianza depositada en mí en la realización de este proyecto y en la oportunidad de formar parte del departamento de Ingeniería Telemática en la realización de la beca.

A mis amigos, que tan buenos momentos hemos vivido, tanto en Cáceres como en Madrid, y, sobre todo, todo lo que nos queda por descubrir.

Gracias a todas aquellas personas que siempre estuvieron en los momentos más importantes de mi vida, y me dieron su cariño desinteresado para que pudiera lograr mis sueños.

Sin vosotros no sería posible.

Resumen

En este documento se explica la herramienta desarrollada para el Trabajo Fin de Grado, cuyo objetivo principal es conocer la calidad de un usuario en Twitter mediante el análisis de una serie de indicadores.

El proyecto de investigación hace uso del parser[1] de Stanford para poder analizar los indicadores lingüísticos, como, por ejemplo, determinar el número de patrones sintácticos distintos en cada cuenta de usuario, y así conocer la variabilidad de sus tweets. Para el caso del registro lingüístico, se utiliza el Wiktionary, donde hacemos uso de una lista de palabras en inglés y su frecuencia de uso para poder determinar el porcentaje de palabras cultas usadas en los tweets. Otros indicadores los hemos denotado como 'generales' y contienen información característica de Twitter, como el número de seguidores y enlaces, haciendo uso de la API de Twitter llamada Twitter4j, y filtrando tweets en inglés con el proyecto Language-detection de Google code. Por último, utilizamos indicadores temporales en referencia a la frecuencia de twitteo.

Se analizan un conjunto de cuentas de diferentes tipos: cuentas de medios de comunicación, cuentas fake, imitadas o parodiadas por las fake, bots[2], cuentas de calidad y usuarios escogidos aleatoriamente, para evaluar los diferentes indicadores considerados y su utilidad para caracterizar tipos de usuarios. Mediante el uso de los box-plots[3], podemos generar gráficos para visualizar el conjunto de valores obtenidos para cada indicador, y poder diferenciar unas cuentas de otras.

[1] Analizador sintáctico.
[2] Programa informático imitando el comportamiento de un humano.
[3] Diagrama de cajas.

Abstract

This document describes a tool developed in the bachelor thesis, the main objective of which is to detect the quality of a Twitter user by analyzing a set of indicators.

The research project uses the Stanford Parser to analyze the linguistic indicators, for example to determine the number of different syntactic patterns in each user account, and then know the variability of their tweets. In the case of linguistic register, the list of English words and their frequency of use from Wiktionary is used to determine the percentage of learned words used in tweets. Other indicators called ‘generals’ contain characteristic information of Twitter, such as followers number and links, using the Twitter API called Twitter4J, and filtering tweets in English using the Language-detection Project of Google code. Finally, temporal indicators are used in reference to the tweeting frequency.

A different types accounts set is analyzed: media accounts, fake, imitated or parodied by fake, bots, quality accounts and randomly chosen users to evaluate different indicators considered and its usefulness to characterize user types. Using the box-plots, graphs to display the set of values obtained for each indicator and be able to differentiate accounts generated.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. Introducción

En este primer apartado se habla de los motivos por los cuales se ha decidido realizar este proyecto de investigación, los objetivos principales y los contenidos de los que consta la memoria.

1.1. Motivación del proyecto

El auge de las redes sociales ha revolucionado el mundo. Surgen como una nueva herramienta de difusión de la información, y permiten la comunicación entre personas sin importar la distancia. Entre ellas destaca Twitter, y el impacto que esta ha generado sobre la sociedad al poder compartir información breve instantánea.

Al circular por la red millones de tweets al día, resulta de interés conocer los usuarios que hacen un uso correcto del idioma, y aquellos que simplemente escriben de manera informal importándoles más el fin que las formas. Es por ello que en este proyecto nos centramos en analizar los tweets de diferentes cuentas de usuarios mediante una serie de indicadores, y comprobar si son suficientes para detectar la calidad, principalmente lingüística, de un usuario en Twitter.

Elegimos Twitter por ser una red social simple y didáctica ya que te permite estar informado al instante de todo lo que te rodea. La mayoría de los usuarios publican mensajes públicos y muy pocos tienen cuentas cerradas, porque la finalidad es dar a conocer tu mensaje y no estar en un círculo privado. Por el contrario existe Facebook, en donde se necesita tener un círculo de amistad para poder comunicarse e intercambiar información. Es por ello que, al tener que analizar un gran número de cuentas, nos decantemos por utilizar Twitter, sin necesidad de tener una lista de amigos previa para analizar distintos mensajes.

La gente puede perder prestigio al escribir con errores lingüísticos o expresiones malsonantes, sobre todo si son personalidades conocidas en un ámbito más intelectual, a pesar del requisito de abreviar el mensaje en 140 caracteres. De ahí nuestra idea de analizar multitud de usuarios, de muy diversos tipos: cuentas de sitios oficiales, cuentas de medios de comunicación, cuentas de usuarios anónimos, cuentas de gente del mundo de la ciencia y la cultura. Pero también mucha información falsa/engañosa: cuentas fake, bots, spam, publicidad, usuarios malintencionados. Se realiza una

recolección para los 6 tipos de cuentas de usuarios de Twitter, la mayoría a través de páginas web con criterio y otras de forma manual, seleccionando 40 cuentas para cada tipo. Para que los valores sean estadísticamente significativos se deben procesar en torno a 50 cuentas, y nunca menos de 30, por lo que hemos elegido 40, debido al gran esfuerzo que implica obtener manualmente cuentas bots (excluyendo aquellas que son conversacionales, las que tuitean titulares de noticias y las que se dedican a retuitear únicamente), haciendo un total de 240 cuentas ya que procesamos 1000 tweets por usuario.

Twitter se ha convertido en una herramienta esencial en la estrategia de las empresas y partidos políticos, por su capacidad de transmitir información casi inmediata. El problema se encuentra en aquellas cuentas que simulan ser usuarios reales y son creadas con un fin determinado, se hacen pasar por personas pero en realidad son programas informáticos destinados a un fin. Se espera que con los resultados podamos diferenciar cuentas de todo tipo, pudiendo conocer, además, la otra cara de Twitter que se aleja del intercambio de información e ideas y se centra en inundar la red con fines inmorales.

1.2. Objetivos

Se trata de desarrollar una herramienta que analice una serie de indicadores que nos puedan servir para detectar la calidad de un usuario de Twitter. Estos indicadores son en su mayoría lingüísticos: basados en el análisis sintáctico de los tweets escritos por un usuario y en el registro lingüístico (variedad de palabras utilizadas en la cuenta). Otros indicadores específicos de Twitter como el número de seguidores, también son interesantes para poder diferenciar diferentes tipos de cuentas, al igual que indicadores temporales referentes al tiempo entre publicación de tweets.

Los principales objetivos del trabajo se especifican a continuación:

- Analizar un conjunto de cuentas de diferentes tipos: periódicos oficiales, cuentas bots, fake, parodiadas por las fake, usuarios normales elegidos aleatoriamente, de manera que a través del análisis de unos determinados indicadores podamos detectar la calidad de un usuario en la red social Twitter.
- Investigar la herramienta para analizar la sintaxis de textos en inglés ya existente, llamada parser de Stanford.

- Identificar a los usuarios que utilizan un registro más culto en Twitter gracias al análisis de una serie de indicadores, generalmente lingüísticos. La idea de usar el Wiktionary, utilizando su lista de palabras en inglés, surge para conocer si un usuario en Twitter utiliza un registro de palabras cultas al twittear, o, sin embargo escribe con abundancia de palabras más comunes, todo ello gracias a la posición de las palabras en el fichero, ordenadas de mayor frecuencia de uso a menor.
- Ofrecer un método para detectar cuentas que son programas informáticos imitando el comportamiento de un humano. Concretamente nos hemos centrado en cuentas bots que no se dedican a retuitear únicamente, no tuitean titulares de periódicos solamente y no son bots conversacionales. Se pretende identificar dichas cuentas, ver cuales se dedican a enviar spam o basura de todo tipo, y que el usuario de Twitter actúe en consecuencia, de manera que se disfrute de un servicio seguro y de calidad.
- Dotar de una herramienta que permita obtener los últimos tweets de cada usuario, leído de un fichero de texto para su posterior análisis, utilizando el API de Twitter. En este trabajo hemos analizado 1.000 tweets por usuario, por lo que hemos procesado un total de 240.000 tweets.

1.3. Contenido de la memoria

El proyecto consta de ocho capítulos donde el contenido se comenta a continuación:

- **Capítulo 1: Introducción**

  En este primer apartado se habla de los motivos por los cuales se ha decidido realizar este proyecto de investigación, los objetivos principales y los contenidos de los que consta la memoria.

- **Capítulo 2: Estado del arte**

  En este capítulo se realiza un análisis de la evolución de las redes sociales, y en particular Twitter, y la relación de nuestro experimento con otros proyectos en la misma línea de investigación. También se describen las herramientas utilizadas para llevar a cabo dicho proyecto.

- **Capítulo 3: Marco regulador y entorno socioeconómico**

Se introducen el conjunto de normas y restricciones que se encuentran en Twitter, los organismos encargados de la seguridad, la privacidad y la protección de datos, y en la influencia que ejercen sobre aplicaciones de terceros que usan datos personales de cuentas de Twitter.

- **Capítulo 4: Desarrollo del trabajo**

  En el cuarto capítulo se explica el desarrollo del proyecto, comentando los módulos implementados, generando el diagrama de componentes y clases, el diseño de la aplicación y los requisitos funcionales y no funcionales.

- **Capítulo 5: Análisis de los experimentos realizados**

  Nos dedicamos a analizar los experimentos realizados para cada uno de los indicadores calculados mediante las representaciones gráficas de los datos. Los agrupamos en tres categorías características: indicadores lingüísticos, generales y temporales.

- **Capítulo 6: Conclusiones y trabajos futuros**

  Este último capítulo pretende presentar brevemente los puntos fundamentales del trabajo desarrollado, mostrar las principales conclusiones obtenidas y discutir las líneas abiertas para posibles investigaciones futuras.

- **Apéndice A: Presupuesto**

  Se realiza un análisis de los costes económicos y temporales en relación a las diferentes fases del desarrollo del trabajo, y la planificación empleada.

- **Apéndice B: Manual de instalación y uso**

  Se proporciona un manual de instalación del software del proyecto, así como su uso para poder obtener los resultados de este trabajo de investigación.

2. Estado del arte

En este capítulo se realiza un análisis de la evolución de las redes sociales, y en particular Twitter, y la relación de nuestro experimento con otros proyectos en la misma línea de investigación. También se describen las herramientas utilizadas para llevar a cabo dicho proyecto.

2.1. Redes sociales

Son comunidades virtuales en Internet que permiten facilitar la comunicación entre gente que se conoce o se desea conocer, y compartir todo tipo de contenidos como fotos y vídeos independientemente del lugar geográfico en que se encuentren. Tienen la ventaja de ser fácilmente accesibles y ha despertado el interés de personas de todas las edades, que se registran y socializan con usuarios de todo el mundo.

En sus comienzos el desarrollo fue lento, donde la información era compartida por pocos usuarios pero a nivel mundial y con una evolución prometedora.

Evolución de las redes sociales, principales sistemas de microblogging[4] y alguna aplicación de mensajería instantánea que revolucionó el mundo.

Figura 1: Evolución de las redes sociales entre 1971-1991

1971: Se envió el primer e-mail entre dos ordenadores situados al lado.

1978: Ward Christensen y Randy Suess crearon el BSS[5] cuya utilidad era informar a sus amigos sobre reuniones, compartir información e incluso publicar noticias.

1979: Es un sistema global de discusión en Internet creado por dos estudiantes de la Universidad de Duke.

[4] es un servicio que permite a sus usuarios enviar y publicar mensajes breves, generalmente solo de texto.

[5] Bulletin Board Systems: sistemas de tablón de anuncios.

1991: El 6 de agosto de 1991, Tim Berners-Lee puso a disposición del público la World Wide Web, que es una red de información donde se puede visualizar contenido multimedia con el uso de hipervínculos[6].

Figura 2: Evolución de las redes sociales entre 1994-2003

1994: Se creó GeoCities que fue un servicio gratuito de webhosting[7], donde permitía a los usuarios crear sus propios sitios web y alojarlos en determinados lugares según su contenido.

1997: Se lanza AOL Instant Messenger, que ofrece a usuarios el chat, al tiempo que comienza el blogging.

1999: Se lanza Blogger, para publicar historias online con una periodicidad muy alta que son presentadas en orden cronológico inverso.

2002: Surge el portal Friendster, donde esta red social de entretenimiento alcanzó los tres millones de usuarios en sólo tres meses.

2003: Esta página fue fundada en julio del 2003, donde sus miembros pueden crear su propio espacio web, personalizándolo con fotos, vídeos, blogs, etc.

Figura 3: Evolución de las redes sociales entre 2003-2006

2003: Nace MySpace en agosto del 2003, en donde los mismos usuarios son los que personalizan su perfil con blogs, fotografías, grupos de amigos, música y vídeos. Uno de sus propietarios era la estrella del pop Justin Timberlake.

2003: Hi5 fue fundada por Ramu Yalamanchi. Al finalizar el año 2007 tenía más de 70 millones de usuarios registrados, 1a la mayoría de ellos en América Latina.

[6] Elemento de un documento electrónico que hace referencia a otro recurso.
[7] Servicio que permite almacenar a los usuarios de Internet contenido accesible vía web.

2004: Facebook nació como un hobby[8] de Mark Zuckerberg, en aquél momento estudiante de Harvard, y como un servicio para los estudiantes de su universidad.

2006: se inaugura Twitter, que es la red social de microblogging, creada en marzo y lanzada en julio del 2006.

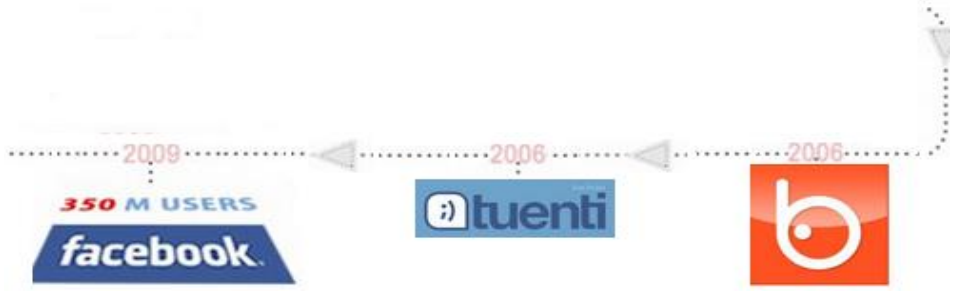


Figura 4: Evolución de las redes sociales entre 2006-2009

Facebook sigue recibiendo ofertas multimillonarias para comprar su empresa. Este mismo año, también comienza su actividad Badoo, una página para buscar relaciones amorosas. A finales del 2006 se lanza Tuenti una red social española propiedad de la empresa Tuenti Technologies S.L, con sede en Madrid y de la cual Telefónica es su accionista principal. Está enfocada al público más joven.

2009: Facebook alcanza los 350 millones de miembros a finales del 2009 y se convierte en la red social más utilizada del mundo con 350 millones de usuarios, adelantando a MySpace. Se crea el servicio Foursquare, basado en localización web aplicada a las redes sociales.

Figura 5: Evolución de las redes sociales entre 2009-2011

2010: Google lanza Google Buzz, su propia red social integrada con Gmail. También se inaugura otra nueva red social, Pinterest, permitiendo a sus usuarios guardar y clasificar por categorías imágenes en diferentes tableros personales. Los usuarios de Internet en este año se estiman en 1,97 billones, casi el 30% de la población mundial. Las cifras son asombrosas: Facebook crece hasta los 550 millones de usuarios: Twitter computa diariamente 65 millones de tweets, mensajes o publicaciones de texto breve. Se crea Instagram, que es un programa o aplicación para compartir

[8] Actividad que se realiza meramente por placer durante el tiempo libre.

fotos con la que los usuarios pueden aplicar efectos fotográficos como filtros, marcos, colores retro, y compartir las fotografías en diferentes redes sociales como Facebook y Twitter.

2011: Se lanza Google+, otra nueva apuesta de Google por las redes sociales. La recién creada Pinterest alcanza los diez millones de visitantes mensuales. Twitter multiplica sus cifras rápidamente y en sólo un año aumenta los tweets recibidos hasta los 33 billones.

Facebook, tras superar los 800 millones de usuarios en todo el mundo, empieza a facilitar datos sobre usuarios activos. Por su parte, Twitter no facilita datos de usuarios desde 2011, año en que rozaba los 200 millones de usuarios en el mundo, de los cuales 100 millones eran activos. Algunos estudios sostienen que a finales de 2012 habría superado los 485 millones de usuarios, de los cuales 288 serían usuarios activos.

Desde su aparición en fase beta hacia abril del 2011, Google Plus ha experimentado uno de los mayores crecimientos en número de usuarios, llegando a los 500 millones a finales del 2012.

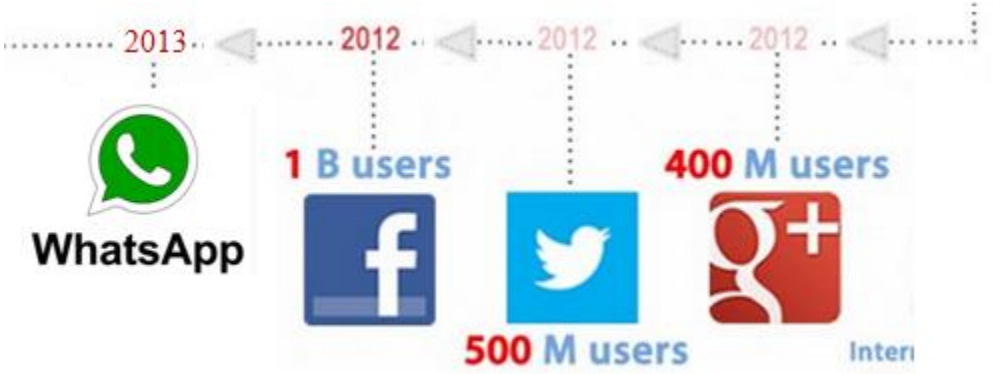


Figura 6: Evolución de las redes sociales entre 2012-2013

2012: Facebook presenta la documentación para salir a bolsa y Twitter genera 12.233 tweets por segundo durante la Super Bowl.

2013: Para diciembre de 2013, el número de usuarios activos en WhatsApp alcanzó los 400 millones. Es una aplicación de mensajería instantánea que permite enviar y recibir mensajes mediante internet de manera económica o gratis. Twitter cuenta con 232 millones de usuarios mensuales activos en el momento de la venta de acciones. Facebook llega a 1,150 millones de usuarios en el mundo.

Las redes sociales están en continua evolución y, por tanto, no existe de momento fecha de fin para la comunicación virtual.

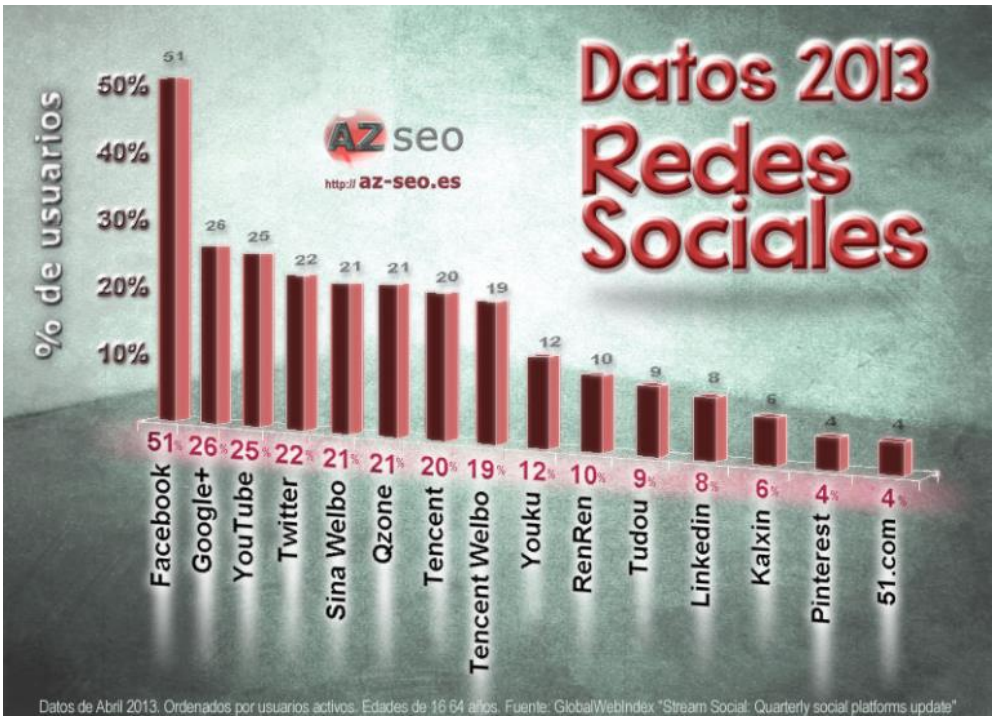


Figura 7: Histograma sobre el porcentaje de usuarios de las principales redes sociales en el año 2013

Cada red social tiene una finalidad. Facebook, Twitter y Google + tienen el propósito de comunicar a sus usuarios y poder compartir información de cualquier tipo, por tanto, son de carácter general, mientras que otras tienen una finalidad más específica, como es el caso de Badoo, que te permite encontrar personas afines a nosotros mediante un buscador inteligente, ya sea gente de tu localidad o de cualquier parte del mundo. LinkedIn es una red social profesional y Youtube está diseñado para subir vídeos y poder compartirlos.

Facebook pulverizó la popular teoría de los seis grado de separación y la situó en menos de cinco. Se trata de una teoría que sostiene que dos personas de cualquier parte del mundo pueden estar conectadas a través de cinco personas que hacen de puntos de unión (en realidad son siete personas, una que inicia la cadena, otra que finaliza y los cinco intermediarios). Dicha teoría defiende que las personas tienen un contacto, y este tiene a otro y este a otro, y así hasta seis que unen a cualquier persona. Con Facebook se ha demostrado que la propuesta de Frigyes Karinthy no solo se cumple sino que se reduce.

Han adquirido tal dimensión las redes sociales que se han convertido en vidas paralelas virtuales, y el fácil acceso a ellas, junto con el desconocimiento del funcionamiento de estas herramientas de comunicación provoca, en muchos casos, situaciones embarazosas o, incluso, de peligro. Otro punto

desfavorecedor de las redes sociales es la pérdida de la privacidad. En el momento que creamos una cuenta nos volvemos accesibles a cualquier persona, por lo que se aconseja no revelar información personal como teléfonos, direcciones y evitar dar cuentas bancarias y contraseñas de cualquier tipo.

2.2. Twitter

Es una red social basada en el microblogging. Fue creada en marzo de 2006 y actualmente cuenta con más de 200 millones de usuarios activos mensuales en todo el mundo, que lanzan en la actualidad unos 500 millones de tweets al día. En los últimos años se ha incrementado notablemente su uso, gracias a la proliferación de nuevos teléfonos móviles integrados con conectividad permanente.

Podemos intercambiar mensajes de texto siempre que no excedan de 140 caracteres, denominados tweets. A medida que escribimos, un contador va indicando cuantos caracteres nos quedan por poner como máximo. Una vez finalizado, pinchamos en Twittear y se envía. Ofrece la posibilidad de comunicación rápida y en tiempo real, que proporciona sensación de cercanía entre la gente a la que comunica.

Iniciación en Twitter

- **Registro**. Nos vamos a twitter.com y creamos una cuenta. El campo más relevante a rellenar será el nombre de usuario, ya que nos identificará en Twitter.
- **Perfil**. Podemos incluir una pequeña descripción de quienes somos, ubicación, subir una imagen, sitio web y conectar con Facebook para publicar tweets en tu perfil o página de Facebook. Existe el nombre de usuario para la cuenta de Twitter y el nombre para que la gente que conoces te reconozca.

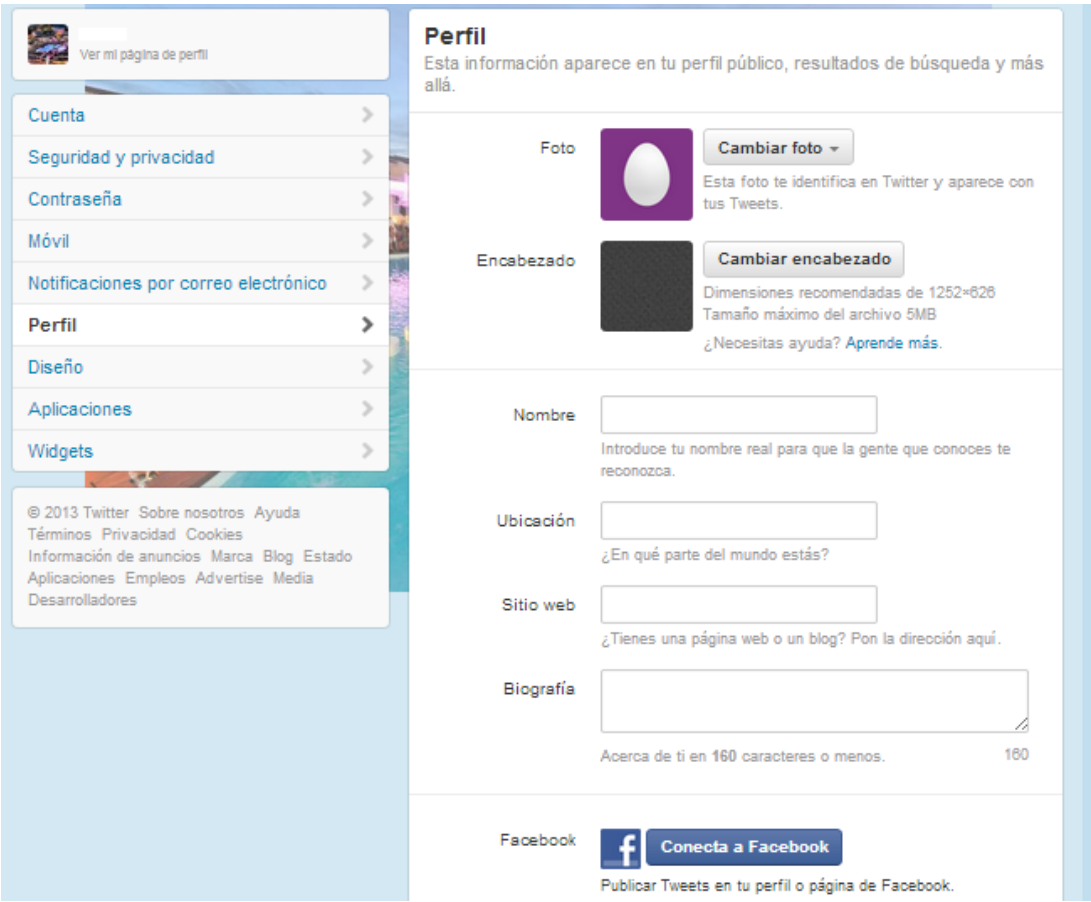


Figura 8: Imagen sobre el perfil de un usuario en Twitter

Mecanismos y utilidades de Twitter

- **Mensajes directos**. Para comunicarse de una forma privada con la gente que nos sigue, debemos seguirles para establecer una conversación, no visible para el resto de la gente.
- **Menciones**. Si queremos referirnos a alguien públicamente, lo hacemos con el formato @nombre_usuario en Twitter. Si a quien mencionamos no nos sigue, es buena forma de despertar el interés de esa persona y que nos siga. Nos puede servir para darnos a conocer y comenzar a construir nuestra comunidad virtual.
- **Hashtags**. Son etiquetas formadas por el carácter # (almoadilla) y una palabra, por ejemplo #TwitterOn. Muy útil para buscar ese hashtag en el motor de búsqueda y aparecerán todos los tweets que hayan usado esa etiqueta.

- **Trending Topics**. Son palabras o frases que más se están usando en ese momento en Twitter, especialmente usados por los medios de comunicación para tener informado a sus seguidores con los temas más actuales. Se puede elegir los TT[9] por país y ciudad.
- **Seguidores**. Son aquellos que nos siguen porque nuestras frases les interesa o por simple amistad. Cuando escribimos un tweet, aparece en su tablón. El hecho de seguir a una persona no implica que dicha persona nos tenga que seguir.
- **Siguiendo.** Indica el número de usuarios que seguimos. Sus tweets aparecerán en la cronología de nuestra página de inicio (más conocido como timeline[10]).
- **Listas.** Sirven para organizar a los usuarios que seguimos por temáticas, intereses en común, etc, de manera que nos resulte más sencillo seguir sus actualizaciones.

A continuación se muestra una imagen de la página principal de una cuenta de Twitter.

Figura 9: Interfaz de Twitter de la cuenta @twitterapi

[9] Abreviación de dos palabras en inglés, Trending Topic, que significa "tema de moda", y expresa la idea de tema del momento.
[10] Historial de todos nuestros tweets, o del conjunto de varios tweets de usuarios clasificados por diversas formas.

Qué hacer con los mensajes que recibimos

- **Responder**. Responder a un usuario después de leer su tweet. Automáticamente se pone el @nombre usuario para que, de esta forma, le llegue nuestra respuesta.
- Marcar como **Favoritos.** Permiten archivar los mensajes que nos parezcan más interesantes.
- **Retwittear.** Es una manera de difundir el mensaje recibido en nuestro tablón entre nuestros seguidores, conservando su contenido. Más conocido como hacer RT[11].

2.3. Trabajos relacionados

Se referencia una serie de artículos con contenidos en la misma línea de investigación que nuestro proyecto.

2.3.1. "Scaling-Laws of Human Broadcast Communication Enable Distinction between Human, Corporate and Robot Twitter Users"

Grabiela Tavares y Aldo Faisal son dos científicos de la Universidad Imperial de Londres que estudiaron las marcas de tiempo de más de 160.000 tweets, desarrollando

un algoritmo capaz de distinguir entre humanos que escriben en su propia cuenta, otras que son gestionadas por otras personas (conocidas como community manager) y cuentas automatizadas llamadas bots que imitan el comportamiento humano [1].

Implementaron dos algoritmos: un clasificador que distingue entre dos y tres categorías de cuentas mencionadas con anterioridad, y un algoritmo que puede predecir cuando uno de estos usuarios enviará su siguiente tweet, centrándose en el análisis de patrones temporales independientemente del contenido de los mensajes.

Describimos en detalle los dos algoritmos utilizados:

1. Clasificador de Bayes que distingue entre dos y tres categorías de cuentas, con una tasa de acierto de 84.6% y 75,8% respectivamente:

[11] Tomar un tweet recibido de alguien a quien seguimos, y re-enviarlo para que nuestros seguidores lo lean.

Con este clasificador tratan de distinguir si tras una cuenta de Twitter se encuentra una persona real, si está gestionada por otra persona (que recibe el nombre de **community manager**) o si son cuentas que simulan ser una persona pero en realidad no lo son (bots).

Una de las maneras utilizadas para la detección de cuentas es mediante los patrones temporales. Hicieron el estudio de la frecuencia relativa de los tweets enviados durante el día, analizando qué franjas horarias son más habituales para el uso de Twitter. Como resultado obtuvieron la siguiente imagen:

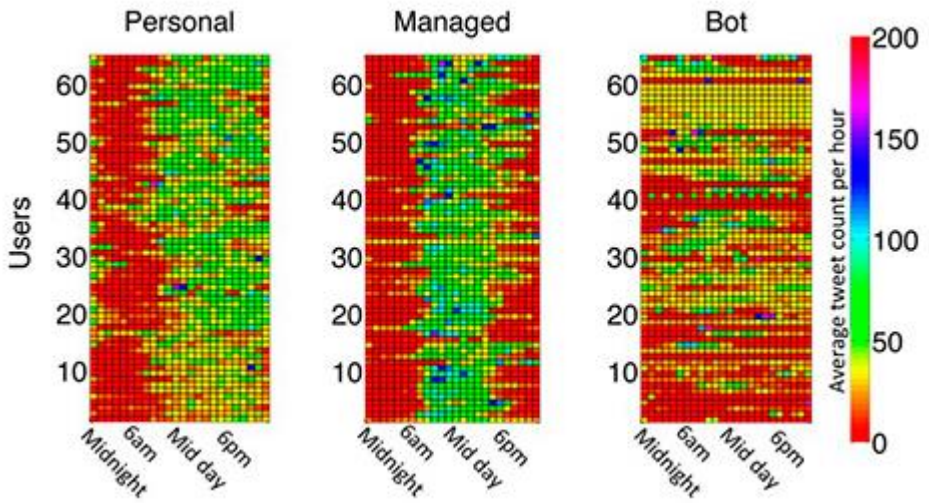


Figura 10: Frecuencia de twitteo en diferentes franjas horarias del día para cuentas bots, community manager y humanos

Se aprecia que las personas twittean en el horario de día, donde se ve que escasean los tweets por la noche al encontrarse dormidos. Por lo tanto, los humanos usan Twitter desde que se levantan (el horario más habitual para despertarse e ir a trabajar suele ser entre las 6am-8am) hasta que se acuestan, de forma intermitente durante ese periodo de tiempo y siguen un patrón de horario que se puede definir en la franja horaria de (6-8) am y las 23:59am. Como siempre hay excepciones, ya que mucha gente no trabaja y algunas lo hacen por la noche, o simplemente trasnochan. Para el caso del gestor de la cuenta de Twitter, su frecuencia de twitteo se centra en su horario de trabajo. Finalmente, los bots twittean durante todo el día sin parar, ya que son un programa informático y no distinguen entre horarios de sueño o trabajo.

2. Algoritmo de predicción del próximo tweet de un usuario en Twiter con $R^2 \approx 0.7$**:**

Crearon un modelo probabilístico para predecir el próximo tweet de un usuario, en base a la distribución del retardo entre tweets de la clase de cuenta de un usuario. Con el anterior clasificador se analizaba el comportamiento colectivo mediante un conjunto de tweets. Este algoritmo utiliza datos de Twitter para estudiar a los usuarios de una forma individual y hace predicciones sobre ellos en la vida real.

Utilizaron la distribución del tiempo entre tweets de cada clase con el fin de generar una función de distribución acumulada correspondiente (CDF). El CDF del retardo entre tweets, denotada como **t**, describe la probabilidad de que un tweet se postee teniendo en cuenta los segundos que han transcurridos desde el último tweet. Dicho retardo se usó para calcular la función de paso, la cual representa la probabilidad acumulada observada de que un tweet se escribirá teniendo en cuenta que han pasado t segundos desde el último tweet. La función de paso se realizó de la siguiente manera:

$$\text{step}(t) = \begin{cases} 0 & \text{if } t < \tau \\ 1 & \text{if } t \geq \tau \end{cases}$$

Figura 11: Función de paso

- τ : es el actual retardo entre tweets.
- t : son los segundos que han pasado desde el último tweet.

Hay dos posibles valores:

- 0, si el tweet ocurrió antes de τ segundos.
- 1, después de τ .

Cada función de paso de comparó a la clase CDF utilizando R^2, que es el coeficiente de determinación encargado de predecir futuros resultados. Se calculó como:

$$R^2 = 1 - SS_{err}/SS_{tot}$$

Figura 12: Coeficiente de determinación

Donde SS_{err} es la suma de los cuadrados de los residuos y SS_{tot} es la suma total de los cuadrados. Entendemos como residuo el valor observado – valor previsto.

En la imagen siguiente, la CDF corresponde a la probabilidad de que un tweet se postee t segundos después del tweet anterior (probabilidad predicha, representada en color rojo), mientras que las funciones de paso corresponden a la probabilidad real de la ocurrencia de tweets (la observada, representada en color azul). Para que ocurra una predicción perfecta para un tweet específico, tendría que coincidir la CDF con la función de paso.

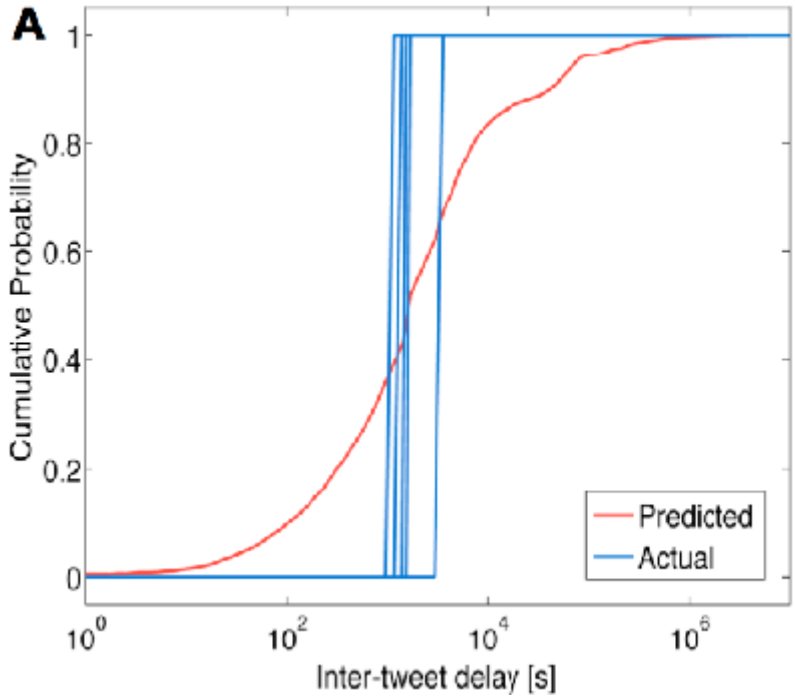


Figura 13: Modelo predictivo

Conclusiones sacadas por los autores del trabajo relacionado

Los resultados obtenidos muestran que se puede distinguir de forma fiable entre las tres categorías de usuarios, así como predecir la distribución del tiempo entre mensajes de un usuario con una precisión razonable. La cola de la distribución del tiempo entre mensajes de los usuarios humanos es diferente de la obtenida para las cuentas administradas y automatizadas. Este resultado es la evidencia de una ley universal que impregna el calendario de las decisiones humanas en comunicación de difusión. Resulta que los humanos como grupos somos más predecibles que los robots.

2.3.2. @spam: The Underground on 140 Characters or Less ∗ Chris Griert Kurt Thomas∗ Vern Paxson† Michael Zhangt

Este trabajo se centra en medir el spam en la red social Twitter. En el experimento se analizaron siete millones de tweets por día utilizando el Streaming API, desde dos fuentes: una captura aleatoria para medir la frecuencia de aparición de URLs y una captura de URLs para analizarlas[2].

Las llamadas listas negras identifican direcciones IP que han enviado spam[12]y las catalogan en una lista pública a fin de informar a todos los interesados. Examinan si el uso de listas negras de URLs ayudaría a frenar significativamente la propagación del spam en Twitter, dada la ausencia de filtrado de spam en Twitter. Se centran en el contenido de los mensajes spam, para conocer las nuevas técnicas de spam utilizadas para atraer a los usuarios, y cuáles son las cuentas involucradas en el envío de spam y su relación con las anteriores.

Otro de los estudios realizados fue generar estadísticas sobre los clics hechos a las URLs que te dirigen a las páginas con spam, gracias al servicio de Bitly que utiliza por defecto Twitter. Bitly también permite acortar las URLs dado el espacio limitado para escribir que ofrece Twitter, lo que permite esconder destinos no deseables.

Resultados:

- Las URLs examinadas permanecen entre 4 y 20 días antes de que sean marcadas como spam. Se trata de un proceso lento en donde el 90% de las visitas ocurren en los dos primeros días de ser posteadas.
- El 8% de 25 millones de URLs analizadas apuntaban a estafas, malware y sitios de phising[13] y figuraban en las listas negras populares.
- Descubrieron que el 0.13% de URLs con contenido spam fueron cliceadas.
- El 8% de los links de los tweets apuntan a URLs de listas negras, 5% es malware[14] y phising, mientras el 95% está orientado a estafas.

Conclusiones:

[12] Correo basura o mensaje basura a los mensajes no solicitados, no deseados o de remitente no conocido, habitualmente de tipo publicitario, generalmente enviados en grandes cantidades que perjudican de alguna manera al receptor.
[13] Método cibernético para estafar y obtener información confidencial de forma fraudulenta.
[14] Software que tiene como objetivo infiltrarse o dañar una computadora o sistema de información sin el consentimiento de su propietario.

- Twitter es una plataforma de gran éxito para coaccionar a los usuarios a visitar las páginas de spam, si se compara con tasas más bajas para el spam de correo electrónico.
- Las URLs maliciosas tardan en ser detectadas por las listas negras de spam y las URLs acortadas favorecen el que se haga clic en ellas.
- Se ha identificado miles de cuentas coordinadas para tuitear URLs que llevan al mismo sitio.
- Algunos RTs fueron comprados a usuarios respetados, también se produjeron secuestros de tweets, en donde a un usuario conocido le cambian la URL y hacen RT.

2.3.3. Who is Tweeting on Twitter: Human, Bot, or Cyborg?

Steven Gianvecchio y Hainning Wang escribieron este trabajo con el objetivo de diferenciar cuentas de Twitter que son humanos, cuentas bots y otras que son cyborg[15].

El notable crecimiento de Twitter ha provocado un aumento de cuentas automatizadas con diversos objetivos, llamadas bots. Este trabajo se centra en bots maliciosos encargados de propagar spam y contenido maligno, al igual que se centra en detectar los llamados cyborgs. Más de 500.000 cuentas de Twitter se utilizaron en la recolección de datos que utilizaron para el estudio. Se realizó un análisis sobre el comportamiento a la hora de twittear, el contenido de cada tweet y las propiedades de la cuenta.

Uno de sus estudios se centró en averiguar cuál de las tres categorías posteaba más tweets diariamente y por hora. Descubrieron que los seres humanos son más activos durante los días regulares de trabajo, de lunes a viernes, y menos durante el fin de semana. Por el contrario, los bots tienen aproximadamente el mismo nivel de actividad todos los días de la semana. Es curioso que los cyborgs son más activos los lunes y disminuyen su actividad de twitteo a medida que avanza la semana, debido al alto nivel de noticias y actividades que circulan en Twitter al comienzo de una semana [3]. En la siguiente ilustración se observa lo comentado:

[15] Cuenta de Twitter en la que un programa tuitea automáticamente, y un usuario humano también tuitea en dicha cuenta.

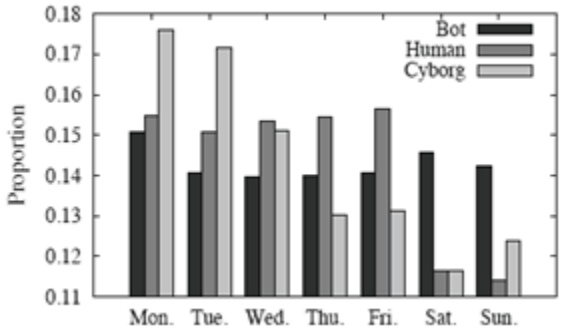


Figura 14: Tweets por día de la semana

Otra de las partes que consta este trabajo es analizar determinadas propiedades de la cuenta de usuario, como contabilizar el número de seguidores y amigos. Distinguen tres grupos:

- I: El número de seguidores es claramente mayor que sus amigos.
- II: Inverso al I.
- III: Los nodos se adhieren alrededor de la diagonal.

En la figura 15 (a) podemos ver al grupo de humanos, donde la mayoría pertenecen al grupo III ya que el número de seguidores es cercano al de sus amigos, pero hay cuentas en las que son mayores los followers al tratarse de celebridades y organizaciones famosas, por lo que se incluiría en el grupo I. La mayoría de las cuentas bots pertenecen al grupo II, ya que se dedican a seguir a todo tipo de usuarios pero poco de ellos les siguen. Sin embargo, se aprecia que algunos bots mantienen el mismo número de followers y followees, porque Twitter impone un límite de proporción de seguidores con sus amigos, precisamente para detectar detectar y eliminar bots. Bots más avanzados hacen unfollow[16] a la gente que sigue si en un cierto tiempo no son seguidos por ellos, de esta forma se mantiene la relación cercana y 1, evitando que sus cuentas sean desactivadas. Los cyborgs, sin embargo, se mantienen en un punto intermedio, en muchos casos poseen un gran número de followers, en otros frecuentan los followees, y en un tercer caso es proporcional la relación.

[16] Dejar de seguir a otro usuario en Twitter.

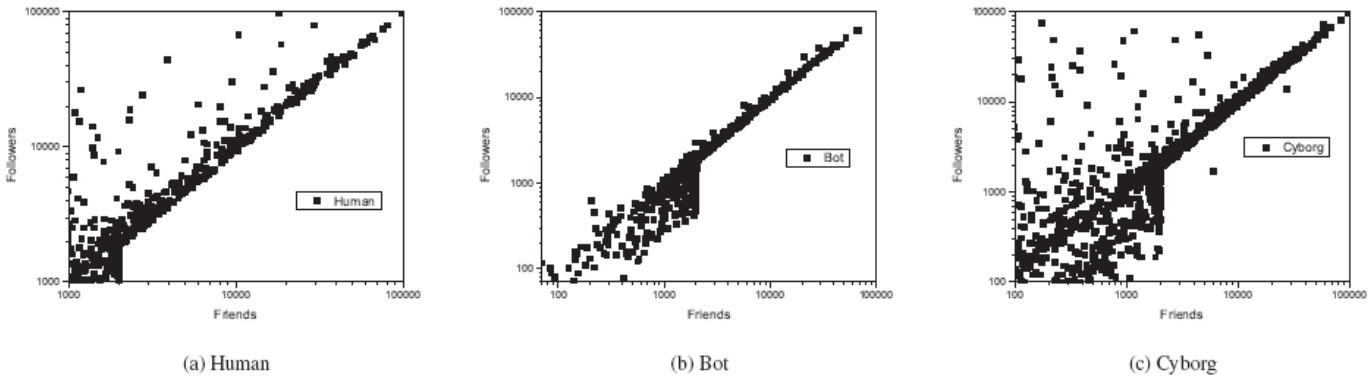


Figura 15: Número de followers and friends

La figura 16 representa la relación entre followers/followees, donde la relación humana es la más alta, ya que los humanos poseen más seguidores en Twitter frente al número de gente que siguen, mientras que las cuentas bots tiene la relación más baja al seguir a multitud de usuarios, pero sin poseer muchos seguidores. Las cuentas cyborgs presentan una relación intermedia, ya que se tratan de seres humanos y, por tanto, tienen una relación proporcional de followers y followees, y al poseer aparatos tecnológicos programados para determinadas funciones, se aprecian valores similares a bots automatizados, de ahí el gran número de gente a la que siguen sin corresponder con el mismo número de seguidores.

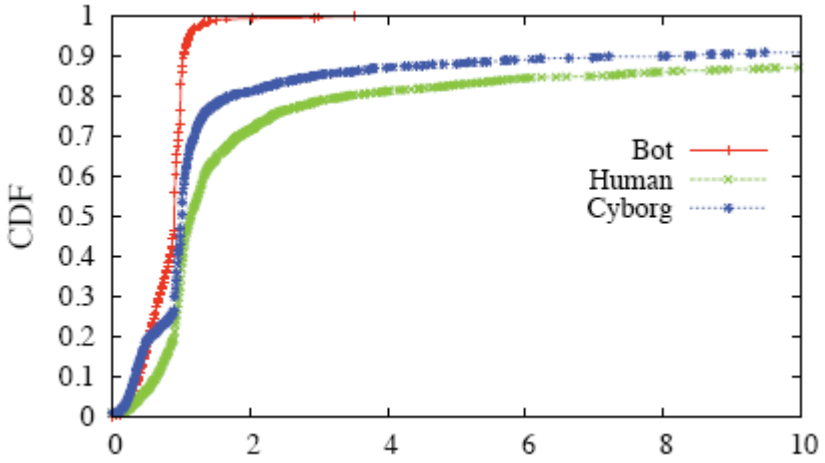


Figura 16: Función de distribución acumulada de la relación entre seguidores y amigos

2.4. Herramientas utilizadas

Para el desarrollo del trabajo se ha utilizado la librería de Java para el API de Twitter para la obtención de los tweets a analizar, el parser de Stanford para el análisis sintáctico de los tweets, el entorno de desarrollo eclipse usando el lenguaje Java para el desarrollo del código, y los box-plots para la representación gráfica de los datos.

2.4.1. Parser de la Universidad de Stanford: un analizador sintáctico

Un analizador de lenguaje natural es un programa que desarrolla la estructura gramatical de las oraciones [4]. Analizadores probabilísticos utilizan el conocimiento de la lengua adquirida analizando frases a mano para tratar de producir la etiqueta gramatical más probable en el análisis de las nuevas sentencias.

El paquete de descarga de la página oficial del parser de Stanford es una implementación, en java, de analizadores de lenguajes naturales probabilísticos.

La versión original de este programa de análisis fue escrita principalmente por Dan Klein, con el código y desarrollo de la gramática lingüística de Christopher Manning. Roger Levy se encargó de la internacionalización, Teg Grenager de la compactación de la gramática y tokenización[17].

No solo existe un parser para analizar textos en inglés, sino que ha sido adaptado para trabajar con otros idiomas. Existe el analizador chino basado en el Treebank chino, que es un corpus[18] lingüístico en el que cada frase ha sido parseada, es decir, anotada con su estructura sintáctica, y para los árabes existe un programa de análisis basado en el Penn Treebank árabe, que anota el texto (en este caso con el idioma árabe) de origen natural para la estructura sintáctica, produciendo un banco de árboles sintácticos. También se ha usado el analizador para otros idiomas, como el búlgaro, italiano y portugués.

Se requiere tener instalado Java 6 (JDK 1.6) o posterior, para ejecutar la última versión. Requiere una cantidad razonable de memoria (por lo menos 100MB, para ejecutar un analizador PCFG[19] con cadenas de hasta 40 palabras de longitud, típicamente, alrededor de 500 MB de memoria para ser capaz de analizar frases largas. Se puede usar tanto en el sistema operativo Windows como en Unix.

El analizador proporciona dependencias de Stanford, las cuales proporcionan una representación de las relaciones gramaticales entre las palabras de una oración [5]. Fueron diseñadas para ser fáciles de entender y para ser utilizadas por personas que quieren extraer relaciones textuales.

[17] Es el proceso de dividir un flujo de texto en palabras, frases, símbolos u otros elementos significativos llamados tokens.
[18] Colección grande de textos escritos por humanos.
[19] Gramática libre de contexto probabilística

Convierte el texto de entrada en un árbol de derivación que puede ser utilizado para su posterior análisis. El análisis sintáctico es el encargado de realizar la verificación de las distintas reglas de formación de un lenguaje, siendo los resultados de dichos análisis representados gráficamente a través de una estructura jerárquica o árbol sintáctico. Es el proceso mediante el cual se pretende lograr la formación del árbol sintáctico, aplicando las reglas de producción del lenguaje.

Gramática libre del contexto [6]

Es la que genera lenguajes libres o independientes del contexto.

Es una gramática formal en la que cada regla de producción es de la forma: V –> w, donde en el lado izquierdo de una producción puede aparecer el símbolo distinguido o símbolo no terminal, en este ejemplo V es el símbolo no terminal, y en el lado derecho de una producción aparece cualquier cadena de símbolos terminales y/o no terminales, siempre de longitud mayor o igual a 1, siendo en este caso w.

Una gramática de contexto libre probabilística se puede entender como:

G = (T, N, S, R, P)

- G representa la gramática del lenguaje.
- T es un conjunto de símbolos terminales.
- N es un conjunto de símbolos no terminales.
- S es el símbolo inicial ($S \in N$)
- R es un conjunto de reglas de producción de la forma $X \rightarrow \alpha$
- P es una función de probabilidad.
    - $P: R \rightarrow [0,1]$
    - $\forall X \in N , \sum P(X \rightarrow \alpha)=1$

S es el símbolo inicial por convención general, pero en procesamiento estadístico del lenguaje natural hay un nodo extra que es el top (ROOT, TOP).

Una gramática independiente de contexto probabilística es igual a una gramática independiente de contexto en la que cada regla tiene asociada una probabilidad. La suma de las probabilidades asociadas a las reglas con un mismo símbolo no terminal en su parte izquierda es 1. Estas gramáticas permiten calcular la probabilidad de una derivación sintáctica a partir de las probabilidades de todas las reglas que se han aplicado.

Ventajas	Desventajas
Permiten dar una idea probabilística de lo buena que es una derivación sintáctica de una frase, permitiendo decidir ante una ambigüedad.	La probabilidad de una frase depende únicamente de la derivación sintáctica y no tiene en cuenta el contexto léxico.
Las reglas probabilísticas se pueden aprender a partir de un conjunto de ejemplos correctamente formados.	Las frases cortas tienen mayor probabilidad que las largas.

Tabla 1: Ventajas y desventajas de la gramática libre de contexto

El parser puede leer diversas formas de entrada de texto plano y dar salida a varios formatos de análisis. Veamos un ejemplo para el parser de inglés que es el que se utiliza en el Trabajo Fin de Grado:

Sentencia de entrada para ser analizada: Sales executives were examining the figures with great care.

Salida:

El siguiente resultado muestra parte de su discurso de texto etiquetado, usando la tabla general con las etiquetas estándar enumeradas alfabéticamente que se utilizan en el Penn Treebank, según el modelo de anotación léxico-gramatical (amalgama) [7]. Formato: palabra de entrada / etiqueta:

Tagging

```
Sales/NNS executives/NNS were/VBD examining/VBG the/DT figures/NNS with/IN great/JJ care/NN ./.
```

Figura 17: Etiquetado de una frase utilizando el parser de Stanford

A continuación muestro una imagen detallada de la palabra, su etiqueta y una pequeña descripción:

Etiqueta	Descripción	Ejemplo
NNS	Sustantivo común plural	Sales
NNS	Sustantivo común plural	executives
VBD	Tiempo verbal pasado	were
VBG	Verbo gerundio	examining
DT	Artículo determinado	the
NNS	Sustantivo común plural	figures
IN	Preposición	with
JJ	Adjetivo numeral indefinido	great
NN	Sustantivo común singular	care
.	Terminador de sentencia	.

Figura 18: Ejemplo explicativo del etiquetado sintáctico de una frase usando el parser de Stanford

Estructura del árbol sintáctico como salida del parser:

Parse

```
(ROOT
  (S
    (NP (NNS Sales) (NNS executives))
    (VP (VBD were)
      (VP (VBG examining)
        (NP (DT the) (NNS figures))
        (PP (IN with)
          (NP (JJ great) (NN care))))))
    (. .)))
```

Figura 19: Árbol sintáctico generado como salida del parser de Stanford

Una forma más cómoda de verlo es realizando nuestro propio árbol sintáctico de derivación:

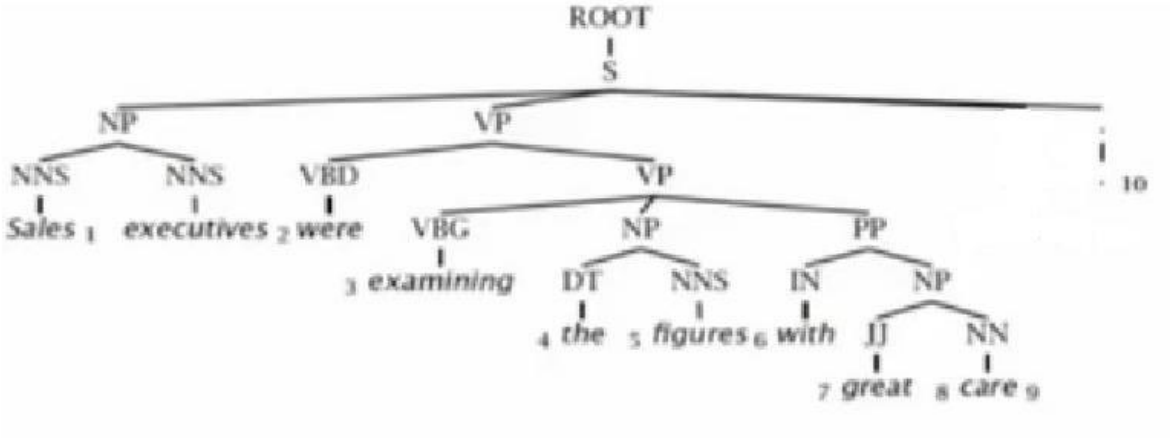


Figura 20: Árbol sintáctico de derivación generado por nosotros

Podemos obtener la siguiente salida que nos muestra el origen y fin de cada etiqueta dentro del árbol para tener una correcta delimitación de etiquetas:

S-(0:10), NP-(0:2), VP-(2:9), VP-(3:9), NP-(4:6), PP-(6:9), NP-(7:9)

Figura 21: Origen y fin del etiquetado sintáctico

Otra salida posible son los tipos de dependencia: las dependencias de Stanford ofrecen una representación de las relaciones gramaticales entre las palabras de una oración.

Typed dependencies

```
nn(executives-2, Sales-1)
nsubj(examining-4, executives-2)
aux(examining-4, were-3)
root(ROOT-0, examining-4)
det(figures-6, the-5)
dobj(examining-4, figures-6)
prep(examining-4, with-7)
amod(care-9, great-8)
pobj(with-7, care-9)
```

Figura 22: Ejemplo usando los tipos de dependencias

Añado una tabla con una pequeña descripción del significado de la etiqueta de cada dependencia gramatical:

Typed dependencies	Description
nn (executives-2, Sales-1)	nn: modificador de nombre compuesto
nsubj (examining-4, executives-2)	nsubj: sujeto nominal
aux (examining-4, were-3)	aux: auxiliar
root (ROOT-0, examing-4)	root: raíz
det (figures-6, the-5)	det: artículo determinado
dobj (examining-4, figures-6)	dobj: objeto directo
prep (examining-4, with-7)	prep: modificador preposicional
amod (care-9, great-8)	amod: modificador adjetival
pobj (with-7, care-9)	pobj: objeto de una preposición

Tabla 2: Ejemplo de relación de dependencias gramaticales

La representación de los tipos de dependencias de Stanford fue diseñado para proporcionar una descripción sencilla de las relaciones gramaticales, en una sentencia que puede ser fácilmente entendible y utilizado por personas sin experiencia lingüística que quieren extraer relaciones textuales.

La siguiente imagen muestra un ejemplo de cómo se implementaría en lenguaje java las tres salidas anteriormente indicadas, donde inicialmente leemos el modelo del parser que nos interesa, después analizamos el String declarado anteriormente y generamos la salida que queremos, en este caso usamos:

wordsAndTags para obtener la salida palabra de entrada/etiquetado.

penn para obtener el árbol sintáctico.

typedDependencies para obtener las relaciones gramaticales entre las palabras de la oración.

Cabe destacar que por defecto se imprime las dependencias colapsadas ya que son más eficientes que las básicas.

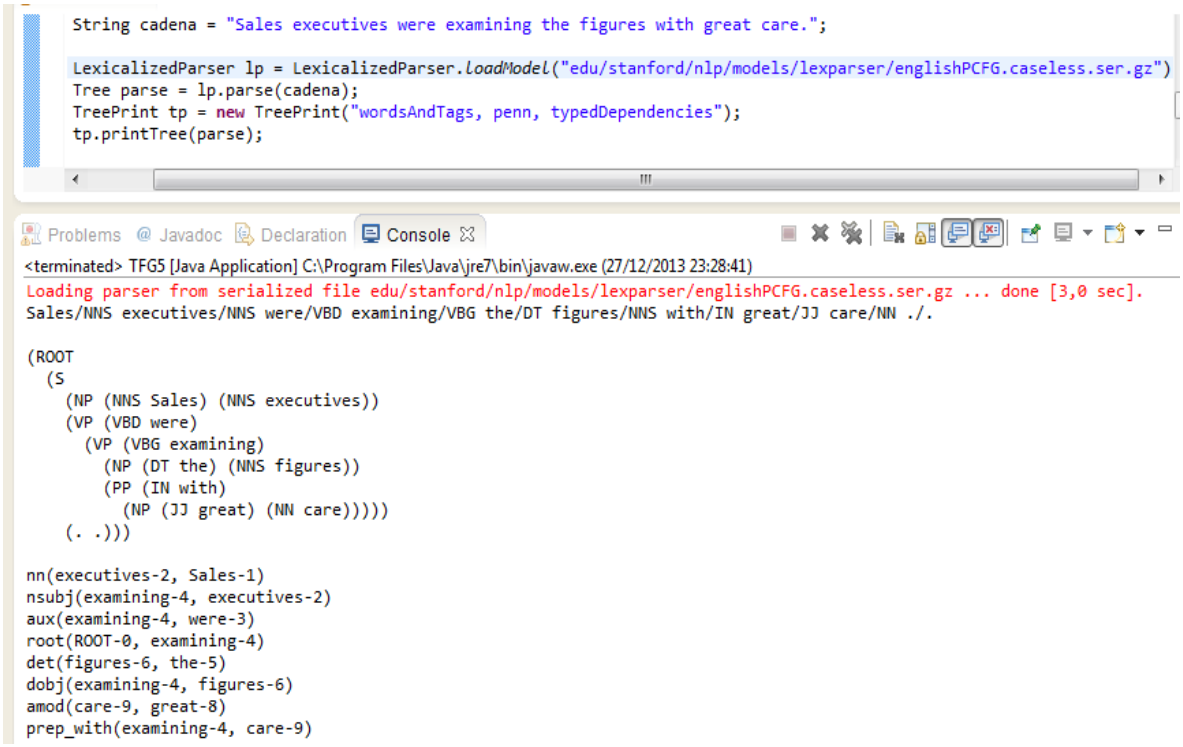


Figura 23: Etiquetado sintáctico, árbol sintáctico y relaciones de dependencias en el entorno eclipse

Otra salida podría haber sido la siguiente (que concuerda con la del código anterior al ser la salida por defecto la dependencia colapsada):

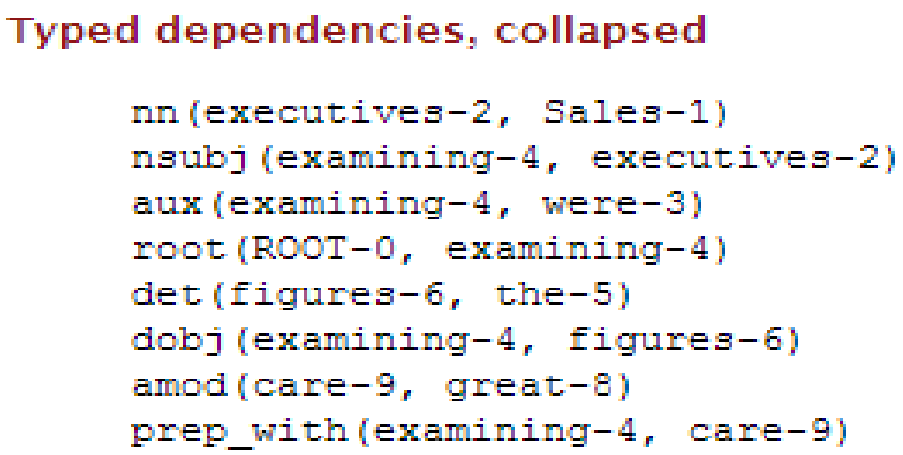


Figura 24: Ejemplo de salida de la dependencia colapsada

En la representación colapsada, dependencias que implican preposiciones, conjunciones, se contraen para conseguir dependencias directas entre palabras de contenido.

Dependencias que implican la preposición “with”:

prep (examining-4, with-7)

pobj (with-7, care-9)

Se convertirá en:

prep_with (examining-4, care-9)

2.4.2. Proyecto Language-detection de Google code

Para la detección del idioma, hemos usado el proyecto Language-detection de Google code, usando la última versión de la librería en java: Language Detection Library (09-13-2011) [8]. El contenido del paquete descargado es el siguiente:

- lib/langdetect.jar, que es la librería en el lenguaje Java y es una de las partes que vamos a utilizar para detectar el idioma del tweet. La otra es mediante la API de Twitter que detallaremos en otro capítulo.
- Directorio profiles, donde se encuentran los perfiles de los idiomas, en concreto 53.
- doc/: referencias a la API de esta librería.
- Es de código abierto bajo la licencia Apache 2.0
- src/: código fuente de esta librería donde hacemos uso de las siguientes clases:
  - DetectorFactory: administra la inicialización y construcciones de Detector.
  - Detector: esta clase detecta el idioma del texto especificado.
  - Language: para almacenar el idioma detectado.
  - LangDetectException: destinada a cazar el error al detectar el idioma.

Probabilidad de la detección de idioma

Se puede obtener la probabilidad de éxito de detección del tweet, con el formato:

```
[en:0.9999962916000225]
```

Figura 25: Ejemplo de salida de la probabilidad del idioma inglés

Como la probabilidad de detección ha dado 0.99 para el idioma inglés, se devuelve toda la lista de probabilidades, y la que tiene el porcentaje de acierto mayor es la que se encuentra en la cima de la lista.

Aquí se encuentra un caso en el que hay una probabilidad de 0.57 de que el texto del fichero se encuentre en francés y una probabilidad de 0.42 de que el texto sea inglés. La decisión de si

continuar con una probabilidad de 0.5 para utilizar dicho tweet o si se necesita por lo menos 0.80, forma parte del desarrollador.

```
C:\Users\Tello\Desktop\TFG\langdetect-09-13-2011>java -jar lib/langdetect.jar --
detectlang -d profiles fichero.txt
fichero.txt:[fr:0.5714272095577031, en:0.42857009344483743]
```

Figura 26: Detección del idioma francés con mayor porcentaje que el inglés

El método getProbabilities() devuelve una lista de idiomas con sus probabilidades. La lista está en orden descendente en cuanto a la probabilidad.

```
public ArrayList<Language> detectLangs(String text) throws LangDetectException {
    Detector detector = DetectorFactory.create();
    detector.append(text);
    return detector.getProbabilities();
}
```

Figura 27: Método para obtener la lista de idiomas con sus probabilidades

Para **comprobar la precisión y el rendimiento** en cuanto a la detección del idioma, se hicieron unas pruebas [9] con los siguientes paquetes:

- Compact Language Detector
- Apache Tika, usando la clase LanguageIdentification, en java.
- Proyecto Language-detection de Google code, en java.

La prueba consistía en detectar el idioma de 1000 textos para cada uno de los 21 idiomas, CLD y language-detection cubren los 21 idiomas, pero Tika le faltan cuatro idiomas: búlgaro (bg), checo (cs), lituano (lt) y el letón (lv), por lo que la prueba se realizó con 17 idiomas que soportan los tres detectores. Al final hubo un total de 17.000 textos. Algunos textos eran de menor tamaño que otros, y también cabe destacar que el CLD detecta al menos 76 idiomas, mientras que el language-detection 53 por lo que su tarea de clasificación es más difícil con respecto a Tika que detecta solo 23 idiomas.

Resultados de la prueba

Paquetes de idiomas	**Precisión**	**Nº detectados correctamente/total**
Language-detection	99,22%	16868 / 17000

CLD	98,82%	16800 / 17000
Tika	97,12%	16510 / 17000

Tabla 3: Resultados de la comparación de tres detectores de idiomas

Con una precisión del 99,22%, la biblioteca de detección de idioma consigue superar a los demás. La CLD consigue un 98,82%, seguido por Tika con 97,12%.

Tika confunde el gallego con el español. Tika y language-detection se equivocan con los idiomas: danés lo confunden con el noruego, esloveno es confundido por el croata y el holandés.

Se comprueba que no todos los detectores se confunden en los mismos casos, por ejemplo, en los textos cortos no siempre fallan los tres. Esto quiere decir que podríamos conseguir una mayor precisión en general, si creáramos una biblioteca de detección de idiomas que combinara las principales características de cada detector.

2.4.3. API de Twitter

Hay disponibles tres tipos de APIs, que se detallan a continuación:

- **Streaming API**. Nos permite recibir el contenido los tweets en tiempo real. El proceso de streaming consigue los tweets de entrada y lleva a cabo cualquier análisis, filtrado, y agregación necesarios antes de guardar el resultado en un almacén de datos. Cabe destacar que los beneficios de tener un flujo de datos en tiempo real hacen que la integración valga la pena en muchos tipos de aplicaciones.
- **REST API**. Ofrece a los desarrolladores el acceso al core de los datos de Twitter. Permite obtener tweets del timeline del usuario y añadir nuevos tweets siempre que la aplicación tenga permisos de escritura.
- **Search API**. Suministra los tweets con una profundidad en el tiempo de siete días que se ajustan a la consulta solicitada, lo que impide que busquemos tweets que fueron escritos con más antigüedad. Se puede filtrar por lenguaje, usuario, localización. A diferencia de las otras dos APIs, esta ofrece una información un poco más limitada del tweet, como por ejemplo, el screen_name, id y url del autor.

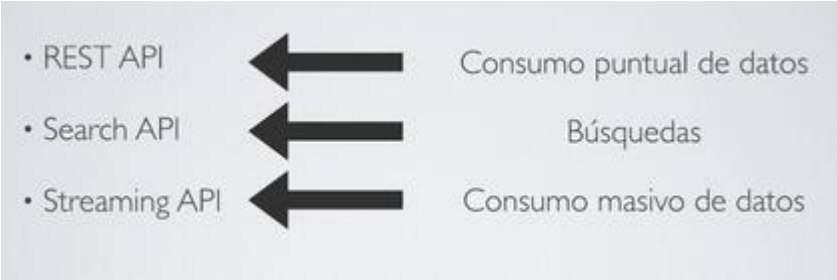


Figura 28: Resumen de las tres APIs

Para utilizar estas APIs debemos utilizar una autenticación OAuth. Oauth es un protocolo de autenticación que permite autenticar a un usuario en una aplicación para actuar en su nombre sin compartir contraseña. Inicialmente no se tiene permisos para acceder a la cuenta del usuario y hay una necesidad de adquirir un token de acceso cuando se redirige al usuario a una URL donde se abre la aplicación. Se obtienen el AccessToken y el RequestToken. Actualmente no se vencen los tokens de acceso. No será válido si el usuario rechaza explícitamente la aplicación de sus ajustes o si un administrador de Twitter suspende su aplicación.

Limitaciones actuales de Twitter [10]**:**

Postear 1000 Tweets por día.
Mensajes directos: 250 por día.
Siguiendo (diario): 1000 por día, pero se pueden obtener tweets en intervalos de media hora donde el límite se distribuye.
Siguiendo (basado en una cuenta): cada usuario puede seguir a 2000 personas en total. Una vez llegados a ese límite, existen límites para el número de usuarios adicionales que puedes seguir, y es diferente para cada usuario. Cuando llegamos a 2000 personas siguiendo, se muestra un mensaje de error por el explorador y hay que esperar hasta que se consiga más seguidores. Por ejemplo, no se puede seguir a 10000 personas si solo 100 personas te siguen.
Cambios de la cuenta de correo electrónico: cuatro por hora.

Los límites de velocidad en la versión 1.1 de la API se dividen en intervalos de 15 minutos, lo cual es una mejora con respecto a la versión anterior que era de 60 minutos. Además, con la nueva versión se requiere autenticación y existe un rango más amplio con las solicitudes [11]:
Peticiones GET: 15 llamadas cada 15 minutos.

A pesar de la ampliación del límite del API, sigue siendo un verdadero obstáculo para los desarrolladores de aplicaciones. La clave es sacar el máximo rendimiento al realizar la paginación de las peticiones.

En Streaming API, el flujo es continuo y la velocidad de recepción de tweets tendrá variaciones que van a depender del ancho de banda de los dos extremos de conexión y la sobrecarga de los servidores de Twitter.

Cuando nos pasamos del límite, el error que nos sale utilizando el REST API en el entorno eclipse es el siguiente:

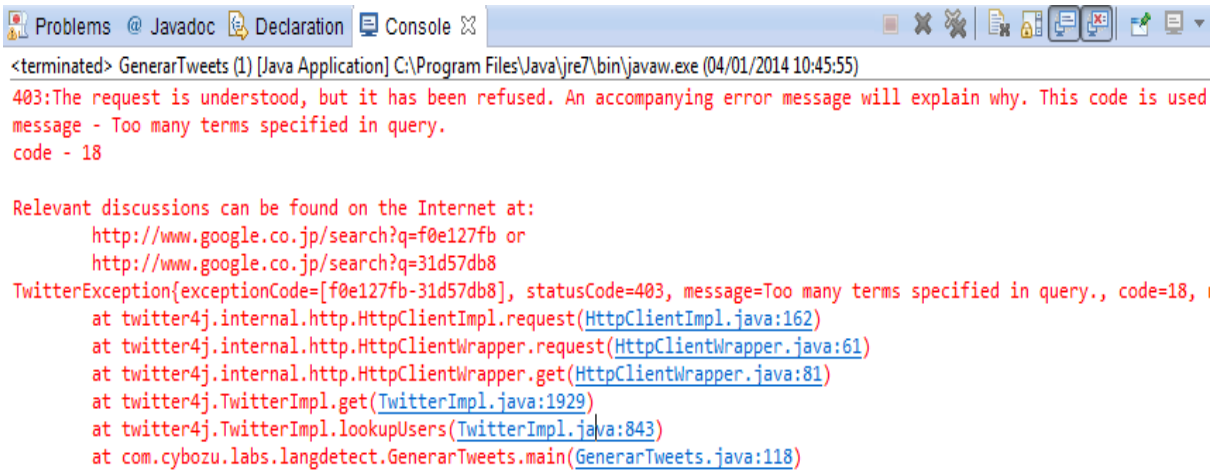


Figura 29: Ejemplo de error por llegar al límite establecido por la API de Twitter

2.4.4. Protocolo OAuth

OAuth es un protocolo de autenticación que permite autorización segura de una API para determinadas aplicaciones, de manera que se proporciona a los usuarios un acceso a sus datos al mismo tiempo que protege las credenciales de su cuenta [12].

Es un estándar abierto lanzada en el 2007 con la idea de que una aplicación web (cliente) pueda acceder a la información de un usuario en otra (proveedor), sin la necesidad de utilizar el nombre de usuario y la contraseña, que sí se utilizaba con el modelo de autenticación básica tradicional.

La red social Twitter usa, como sistema de autenticación de su API, el protocolo OAuth. El proceso de autenticación lo explicamos a continuación, aunque será en el apartado 4.5.1 donde se detalla el proceso de registrar nuestra aplicación en Twitter, y todo lo desarrollado para ello, incluyendo la parte del protocolo OAuth [13]:

- Adquirir el "request token" y "request secret", que son las credenciales del cliente, al finalizar el registro de la App en Twitter.
- Se necesitan permisos para acceder a la cuenta del usuario, y es necesario adquirir un token de acceso, solicitando la autorización redireccionando al usuario a la página de autorización, ofreciendo un número de pin para generar un fichero con la clave.
- Una vez adquirido las claves del usuario, la aplicación ya tiene autorizado por parte del usuario a acceder a sus recursos, sin ser necesario el intercambio de usuario y contraseña.

2.4.5. Java

Java es un lenguaje de programación orientado a objetos y basado en clases, siendo uno de los lenguajes de programación más populares en uso [14]. Es la primera plataforma informática creada por Sun Microsystems en 1995.
Se puede descargar gratuitamente en[20].

Figura 30: Logo de Java

Características [15]**:**

- Es robusto, ya que no hay que preocuparse de punteros, fugas de memoria. El sistema de Java maneja la memoria del ordenador por nosotros.

[20] http://java.com/es/

- Es portable, debido a que como el código compilado de Java es interpretado, un programa compilado de Java se puede utilizar por cualquier computadora que tenga implementado el intérprete de Java.
- Es independiente de la arquitectura: el código resultante de compilar un programa en Java es interpretado por diferentes computadoras de la misma manera, solo hay que implementar un intérprete para cada plataforma. El que realmente ejecuta el código generado por el compilador no es el procesador del ordenador directamente, sino que es gracias a una máquina virtual.
- Es distribuido: facilita clases para su uso en aplicaciones de red, lo que permite establecer conexiones entre un servidor que escucha todo el rato, y un cliente que hace peticiones, a través de sockets, de forma que facilita la creación de aplicaciones distribuidas.

2.4.6. Eclipse

Es una plataforma de desarrollo de código abierto basada en Java. Es un marco y un conjunto de servicios para construir un entorno de desarrollo a partir de componentes conectados (plug-in) [16].
Es de código abierto y se usa para desarrollar "Aplicaciones de Cliente Enriquecido".
Ventajas [17]:

- El **IDE**[21] emplea módulos (plug-in) para proporcionar toda su funcionalidad al frente de la plataforma del cliente, sin necesidad de incluir todas las funcionalidades aún sin necesitarlas.
- Permite interactuar con bases de datos, trabajar con diferentes lenguajes de programación como C/C++ y Python. El plug-in más conocido es para el desarrollo de Java llamado **JDT**[22].
- Uso sencillo en la compilación de programas, sin necesidad de utilizar la línea de comandos que a veces resulta más lioso.

[21] Entorno de desarrollo integrado

[22] Java Development Tools (herramienta para el desarrollo de Java)

Figura 31: Logo de eclipse

Figura 32: Logo de Eclipse

2.4.7. Twitter4j

Es una librería de Java no oficial para la API de Twitter. Se puede integrar nuestra aplicación de Java fácilmente con el servicio de Twitter. Gracias a Twitter-4j podemos obtener la lista de los últimos tweets del timeline de un usuario, enviar y recibir mensajes directos, buscar tweets [18].

Características:

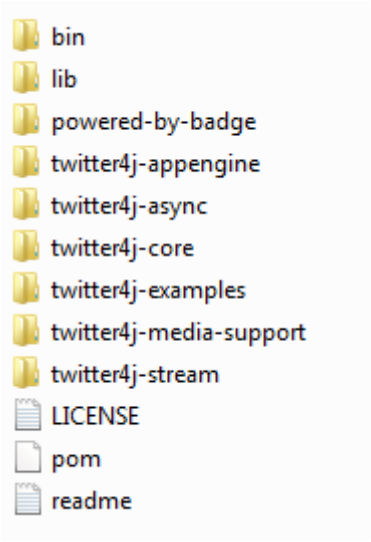
- Utiliza el lenguaje popular Java y funciona con la versión 5 o más.
- Soporta OAuth.
- Totalmente compatible con la versión actual TwitterAPI 1.1
- Disponible para Android y Google App[23] Engine (que permite crear y alojar aplicaciones web en los mismos sistemas con los que funcionan las aplicaciones de Google.

Requisitos del sistema:

- OS: Windows
- JVM: Java 5 o posterior

En la página oficial nos descargamos twitter4j-3.0.6-SNAPSHOT.zip (javadoc), que contiene la siguiente estructura de directorios:

[23] Aplicación de software



Usamos twitter4j-stream-3.0.6.jar para monitorizar los últimos tweets usando el Streaming API (para encontrar usuarios aleatorios que estén twitteando en ese momento) y twitter4j-media-support.3.0.6.jar para obtener los últimos 1400 tweets sincronizados de un usuario escogido de forma predeterminada.

Posibles errores

Entre los errores más comunes que pueden surgir a la hora de utilizar el API Twitter4j se encuentran:

- **Límite de consultas realizadas (código HTTP 403).** El API limita una cantidad de consultas posibles, y, en caso de sobrepasarla, nos darán este error. Como solución nos queda esperar un tiempo especificado en el API para volver a realizar peticiones.
- **Acceso no autorizado (código HTTP 401).** Se manifiesta si durante el proceso de obtención del token de acceso ocurre algún fallo.
- **Información no encontrada** (error 404).
- **Twitter no disponible o saturado** (error 503).

2.4.8. Wiktionary

Es la contracción de wiki y diccionario que en inglés se dice Wiktionary. Se trata de un proyecto de diccionario libre de la Fundación Wikimedia donde se colabora para producir un diccionario

multilingüe de contenido libre. La finalidad es describir todas las palabras de todas las lenguas que usan descripciones, definiciones, traducciones y sinónimos [19].
Se puede editar, y todo el contenido es dual-licensed bajo Creative Commons Attribution-ShareAlike 3.0 Unported License como por GNY Free Documentation License.

Hacen un recuento de las palabras más relevantes en un idioma, también contabilizan lemas. Por ejemplo, en el caso del idioma inglés, el verbo ser está representado por las conjugaciones "is", "are", "were", etc.
Hay varias listas, siendo la más actual la lista de frecuencias a partir del 16 de junio del 2006.
Hay un total de 36663 palabras en inglés [20]. El formato que aparece en la página web es el siguiente:

- Número de palabra
- Palabra
- Frecuencia de uso

Se encuentran ordenadas de mayor frecuencia a menor frecuencia, lo que permite conocer las palabras de uso más corriente, y aquellas que son de un registro más culto. La mayoría aparecen en el Proyecto Gutenberg, que es un voluntario esfuerzo por digitalizar y archivar obras culturales, para fomentar la creación y distribución de libros electrónicos. En este caso, se descargó la lista de libros en julio de 2005.

a · about · after · all · and · any · an · are · as · at · been · before · be · but · by · can · could · did · down · do · first · for · from · good · great · had · has · have · her · he · him · his · if · into · in · is · its · it · I · know · like · little · made · man · may · men · me · more · Mr · much · must · my · not · now · no · of · on · one · only · or · other · our · out · over · said · see · she · should · some · so · such · than · that · the · their · them · then · there · these · they · this · time · to · two · upon · up · us · very · was · were · we · what · when · which · who · will · with · would · you · your

Figura 33: 100 palabras más comunes de la lista en inglés del Wiktionary

2.4.9. Box-plot

¿Qué es?

Un diagrama de cajas es un gráfico que visualiza un conjunto de datos basándose en cuartiles. Nos muestra información sobre los cuartiles Q1, Q2, Q3 y valores atípicos en una "caja" (rectángulo) y "bigotes" (dos brazos).

Elaboración [21]**:**

- La caja central nos indica el rango en el que se concentra el 50% central de los datos.
- Los extremos de la caja equivalen a los cuartiles Q1 (25% de los datos) y Q3 (75% de los datos) de la distribución.
- Dentro de la caja existe una línea central que se llama la mediana. Si la variable es simétrica, la línea se situará en el centro de la caja. Q2 es el cuartil que la representa.
- Existe el rango inter cuartílico (RIC), que es la diferencia entre el tercer y el primer cuartil, calculado como: Q3 – Q1.
- Las líneas que se extienden desde la caja se denominan "bigotes". Para representarlos, necesitamos calcular los límites inferior y superior, Li y Ls respectivamente, necesarios para identificar los valores atípicos, los cuales están distantes del resto de los datos y serán aquellos que estén fuera del intervalo (Li,Ls) y se representan con un círculo. Se calculan así: Li = Q1 – RIC*1.5 y Ls = Q3 + RIC*1.5.
- Existen casos en que los valores los podemos considerar como extremadamente atípicos, en cuyo caso excederán de: Q1 – 3 * RIC o Q3 + 3* RIC. Se representan con un asterisco.

Uso

- Son apropiados para representar variables que presentan bastante desviación de la distribución normal.
- Sirven para comparar gráficamente el comportamiento de una variable en distintos grupos.
- Cómodo para detectar valores atípicos (outliers).
- Los box-plot pertenecen a las herramientas de la estadística descriptiva, la cual se dedica a analizar y representar un conjunto de datos, con el fin de escribir apropiadamente las características de este.
- Permiten ver la dispersión de los valores con la mediana, y así poder diferenciar un tipo de cuentas de otras, como es la finalidad de este Trabajo Fin de Grado.

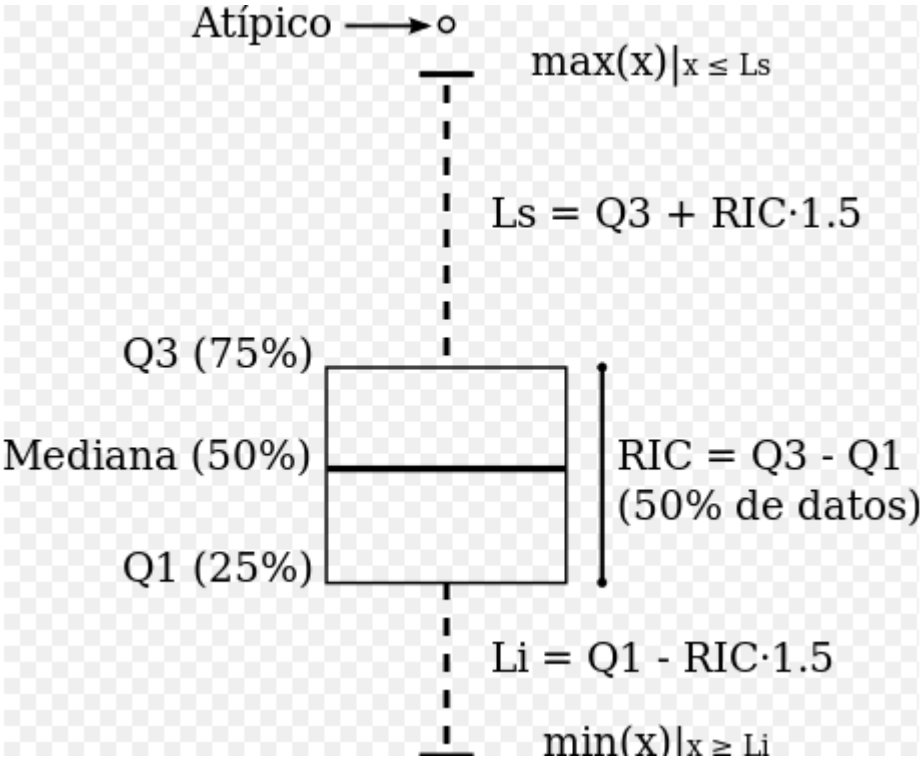


Figura 34: Diagrama de caja (Box-plot)

2.4.10. Entorno R

R puede entenderse como un lenguaje de programación, como un potente software de análisis estadísticos o incluso como un generador de gráficos. Es de uso libre en el dominio público, y surge del esfuerzo cooperativo de personas e instituciones académicas relevantes relacionadas con la Estadística y la Computación en todo el mundo. Permite trabajar con una ventana de interacción con el usuario, llamada R Commander, facilitando el uso de R como un software para el análisis de datos [22].

R posee capacidades de análisis estadístico, y también es un buen generador de gráficos. Existe la posibilidad de cargar diferentes bibliotecas o paquetes con finalidades específicas de cálculo o gráfico. Se distribuye bajo la licencia GNU GPL, disponible para los sistemas operativos GNY/Linux, Windows, Unix y Macintosh.

3. Marco regulador y entorno socioeconómico

En este capítulo vamos a comentar la seguridad y privacidad de Twitter en general, al igual que el sistema de protección de datos que presenta, y luego nos centraremos más detenidamente en cómo influye la ley orgánica de protección de datos a la hora de desarrollar una aplicación basada en Twitter, y, en particular, en nuestra aplicación, que se centra en la recogida de información de Twitter como puede ser la captura de tweets, fecha del tweet y número de seguidores entre otros.

3.1. Seguridad y privacidad general de Twitter

A los menores de 13 años no se les permite el registro en Twitter. Si la comparamos con otras redes sociales como, por ejemplo, Facebook, recopila poca información del usuario, por lo que los problemas relativos a la seguridad y privacidad dependerán en gran parte a lo que twittee más que la información personal rellenada por el usuario. Obligatoriamente será necesario un nombre de usuario, contraseña y dirección de correo, el resto son opcionales tales como la ubicación, sitio web.

Todo lo que publiquemos en Twitter será visible para todos a menos que protejamos nuestros tweets, en cuyo caso solo los usuarios aceptados podrán visualizar tu perfil, de forma que los tweets publicados en el pasado estarán visibles en algunos sitios y los del futuro no estarán disponibles públicamente. Es totalmente desaconsejable proteger los tweets porque una de las finalidades es difundir los tweets al máximo número de personas, y la privacidad de ellos hará que muchos twitteros no te sigan. A veces es recomendable crearse dos cuentas, una para el desarrollo profesional y que sea público para dar a conocer tus ideas y promocionar tu empresa o lugar de trabajo. Y otra para uso privado, donde tienes más libertad para comentar temas de actualidad en un entorno de amistad, protegiendo los tweets al ser más personales.

Figura 35: Opción de proteger los tweets en Twitter

Consejos y mejoras

- Se recomienda **no** marcar la opción de mostrar la **ubicación de los tweets** para mayor seguridad.
- Antes del 14 de febrero del 2012, Twitter añadió la opción de habilitar las conexiones HTTPS, lo que permitió que los datos viajaran encriptados entre el ordenador y los servidores de la red social Twitter, mucho más difícil de interceptar. Ha pasado de ser optativo a implementarse por defecto para todos los usuarios, lo que es realmente seguro porque se **protege la información**. Con esta mejora, será más seguro Twitter en una conexión a internet sin protección como una red Wi-Fi pública.
- Se puede reestablecer la contraseña simplemente con el nombre de usuario y la contraseña actual. Para una mayor seguridad, se puede complicar un poco más el proceso de recuperar la contraseña introduciendo el número de teléfono o el correo electrónico, ya que es información adicional más difícil de conocer que el nombre de usuario.
- Una de las mejoras que introduce Twitter es el uso de la verificación del inicio de sesión. En lugar de tener una sola contraseña, se introduce una nueva contraseña para garantizar que eres tú el que realmente te conectas a Twitter, recibiendo un mensaje de texto al teléfono móvil aceptando la siguiente casilla:

Figura 36: Casilla en Twitter para verificar el inicio de sesión con el teléfono

- Se puede **conectar** la cuenta de **Twitter con Facebook**, de forma que los tweets se publican automáticamente en el muro de Facebook. Se puede configurar quien puede ver esos tweets en tu muro de Facebook. Las respuestas y mensajes directos no serán publicados. También se puede conectar la cuenta de Twitter a una página de Facebook. Hay que tener especial cuidado en lo que twitteamos, ya que las personas que nos siguen en Twitter no tienen por qué ser las mismas que tenemos como amigos en Facebook. Por esto y por tweets desafortunados, se nos da la opción de eliminarlo en cualquier momento.

Figura 37: Ejemplo de tweet que se encuentra conectado a Facebook

- Ya sea por el vicio que podamos tener con la red social Twitter, que a veces se convierte en un problema llamado adicción a las redes sociales, o porque simplemente nos hemos cansado, nos han robado la cuenta, si tenemos cualquiera de estos problemas, podemos **desactivar la cuenta de Twitter** en_ configuración->cuenta->desactivar cuenta. Tenemos 30 días para recapacitar y volver a iniciar sesión con todos nuestros datos activos, una vez pasado este tiempo, la cuenta se eliminará.

Figura 38: Mensaje antes de desactivar la cuenta de twitter

- Solo se puede enviar mensajes directos a personas que te siguen y puedes recibir mensajes directos de usuarios a los que sigues.
- Como mejora de la privacidad se encuentra la opción de que otra gente te encuentre en Twitter a través del correo electrónico.
- Puedes permitir a Twitter mostrar anuncios sobre las cosas en las que mostraste algún tipo de interés. Se trata de anuncios personalizados basados en información compartida por socios de publicidad.

Desventajas

- Twitter utiliza como medida de privacidad la opción de bloquear a un usuario, de modo que si te bloquean no puedes seguir a esta persona, no puedes retuitearla, pero en cambio si puedes ver su perfil, y si tiene sus tweets públicos puedes ver todo lo que twittea, lo que resulta poco privado si lo comparamos con Facebook, que si bloqueas a alguien en Facebook, a no ser que le desbloquees, no podrá acceder a tu perfil desde la cuenta bloqueada.

3.2. Ley Orgánica de Protección de Datos y la API de Twitter

En esta parte vamos a analizar con la perspectiva de la Ley de Protección de Datos (LOPD) y la Política de Protección de Datos de Twitter [23], las consecuencias legales de desarrollar aplicaciones basadas en la API de Twitter.

3.2.1. Introducción a la Ley de Protección de Datos (LOPD)

LOPD son las siglas que hacen referencia a la Ley Orgánica del 13 de diciembre de 1999, de Protección de Datos de Carácter Personal, que es la norma jurídica que, desde el 14 de Enero del año 2000, regula en España todo lo relativo al tratamiento de los datos personales de las personas físicas. Establece cómo se debe deben recoger, tratar y ceder los datos de carácter personal para no perjudicar con ello los derechos de sus titulares [24].

Con una ley aprobada hace 15 años es difícil conocer cómo puede influir LOPD en el momento de desarrollar una aplicación basada en la API de Twitter. Las nuevas tecnologías van un paso más adelante que la legislación de privacidad y la LOPD, y será difícil que sean alcanzadas.

Con el uso de la API de Twitter se pueden desarrollar aplicaciones de diferente grado de interacción con los usuarios. En función de la API utilizada se presentarán diferentes situaciones legales, que se dividen en: REST API, destinado a obtener actualizaciones del perfil y consultas como el número de followers, etc, junto con el Streaming API para monitorizar el timeline, y Search API para una búsqueda en el tiempo de siete días.

3.2.2. Recogida y uso de la información

Twitter recoge información de los usuarios para poder mejorar continuamente los servicios que ofrecen, siempre protegiendo los datos y mostrando al público únicamente lo que el usuario ha permitido en su configuración de cuenta.

Al registrarnos en Twitter, rellenamos información personal. Algunos de estos datos se hacen públicos como el nombre de usuario, incluso se puede encontrar realizando una búsqueda sin estar registrado. En la configuración adicional, se puede facilitar datos como el teléfono móvil, de modo que Twitter se puede poner en contacto con nosotros a través de SMS e incluso al correo electrónico.

La necesidad de expansión en todo el mundo hace que de forma predeterminada todo sea público, ya sean los tweets favoritos, seguidores, tweets, pero siempre con posibilidad de hacerlo privado. Constantemente hacen estadísticas con los usuarios e intentan mejorar sus servicios. Al ofrecer tu ubicación te pueden mostrar información local, anuncios sobre tu zona. Twitter utiliza el programa Bitly, aparte de recortar los enlaces, se puede contabilizar hasta el número de clics que ha conseguido un link.

Para que un usuario no tenga que estar introduciendo en cada página del servidor su usuario y contraseña, existe lo que se denomina cookie, que es una pequeña información que se guarda en el ordenador del usuario y en cualquier momento la página puede pedir información, que contiene la cookie, al ordenador del usuario. De forma predeterminada los navegadores aceptan cookies, puedes cambiar la configuración, aunque existe la posibilidad de que dejen de funcionar. Cada vez hay más intentos de programas espías en el robo de cookies, pero el hecho de que Twitter utilice https, la información va encriptada.

Twitter posee los datos de registro de cada usuario como pueden ser su IP, sistema operativo, cookies, ya que los usuarios interactúan con sus servicios al visitar determinadas páginas webs, usan la aplicación de Twitter para obtener los tweets del usuario, número de seguidores e infinidad de información.

En la recogida de datos de cada usuario en Twitter, utilizando la API de Twitter para el desarrollo de cualquier aplicación, diferenciamos tres tipos [25]:

Datos obtenidos de Twitter "en abierto". Al tratarse de una red abierta donde podemos recoger información del timeline público de un usuario en Twitter, utilizando una aplicación y sin la participación del propio usuario, resulta muy cómodo para los desarrolladores. La Ley de Protección de Datos ha recalcado que solo se consideran fuentes de acceso público los diarios, boletines oficiales y medios de comunicación, e Internet no lo considera un medio de comunicación, y que, por lo tanto, no es una fuente de acceso público según recuerda la Agencia Española de Protección de Datos (AEPD). Por tanto, es ilegal la configuración de un fichero o base de datos desde Twitter sin el consentimiento del usuario afectado, pero mostrar de maneras diferentes su información es legal.

Datos que la aplicación pueda obtener directamente del usuario, usando un paso previo para darse de alta como puede ser una contraseña o un correo electrónico, o datos que se soliciten de manera obligatoria o voluntaria como parte del proceso de darse de alta. Este caso es muy común, donde se dará de alta un fichero en el Registro General de Protección de Datos y una serie de obligaciones que son objeto de ley ya sea políticas de seguridad y privacidad, obtener el Documento de Seguridad, atendiendo a las condiciones elegidas por el usuario en el proceso de alta anterior en cuanto al uso de su información.

El usuario da permiso para conectarse a su cuenta de Twitter y obtener sus datos. Según la Política de Protección de Datos de Twitter [24], en el apartado "Cesión y revelación de información" se dan a conocer determinadas situaciones en las que Twitter puede revelar datos personales del usuario, resumidas a continuación:

- Mediante el **consentimiento** del propio usuario, que autoriza a un cliente o una aplicación web de un tercero a acceder a su cuenta de Twitter para usar su información.
- Información que **no** es de **contenido privado**, como la lista de seguidores, tweets públicos.
- **Normas imperativas y daños**. Se revelará o conservará información para el cumplimiento de una ley, como puede ser un fraude, requerimiento legal; proteger la seguridad de un

usuario en Twitter. No se pretende limitar las defensas u objeciones que se pueda tener contra una aplicación de un tercero que accede a tu cuenta.

- Se puede compartir datos personales a proveedores de servicios por obligación de confidencialidad, siempre y cuando los usen bajo determinadas condiciones.

4. Desarrollo del trabajo

Se comentan los módulos que componen la arquitectura de nuestra aplicación, se describe el diseño elegido y los requisitos funcionales y no funcionales a tener en cuenta para utilizar la aplicación. Finalmente se detalla el diagrama UML de las clases que componen el proyecto, así como sus atributos y métodos.

4.1. Arquitectura del proyecto

El proyecto consta de una arquitectura bien definida en la que todos los módulos mantienen relación entre sí. A continuación se presenta el diagrama completo del proyecto, explicando cada punto de manera general, y entraremos en profundidad en el apartado 4.5.

Módulo 1:

1. **Registro de una App en Twitter.** Registramos nuestra aplicación para tener acceso a la cuenta de Twitter.
2. **Fichero de claves de la aplicación**. Lo necesitamos para poder leer esas claves y generar el fichero twitter4j.properties.
3. **Fichero con las claves del usuario**. Es necesario para la autenticación por Oauth, de modo que se puede pedir recursos al servidor por estar autorizado como pueden ser la obtención de tweets.

Módulo 2:

4. **Búsqueda de cuentas de Twitter**. Las dividimos en 6 tipos mostrando algunos ejemplos: cuentas de periódicos (@nytimes), bots (@Favstar_bot), verdaderas (@Barack_Obama), simuladas (@ThePressObama), de calidad (@HeremyHL) y las aleatorias.
5. Insertamos todas las cuentas en el fichero todas_cuentas.txt, para posteriormente leerlas todas.

Módulo 3:

6. **Generación de tweets**. Usamos la librería Twitter4j de Java, donde hacemos peticiones a Twitter utilizando el API REST.
7. Creamos **dos ficheros** por cada usuario, dentro del directorio cuentas.
8. **Fichero con los tweets en cuestión**. Este fichero nos interesa para generar únicamente los tweets, y más adelante si queremos cualquier otra información la almacenamos en otro fichero diferente.

9. **Fichero con información relativa a cada tweet**. Se encuentra ligado al fichero anterior, donde cada línea es información específica del tweet.

   Fichero log. En caso de ocurrir algún error en cualquiera de los dos ficheros, se guarda la información detallada del problema.

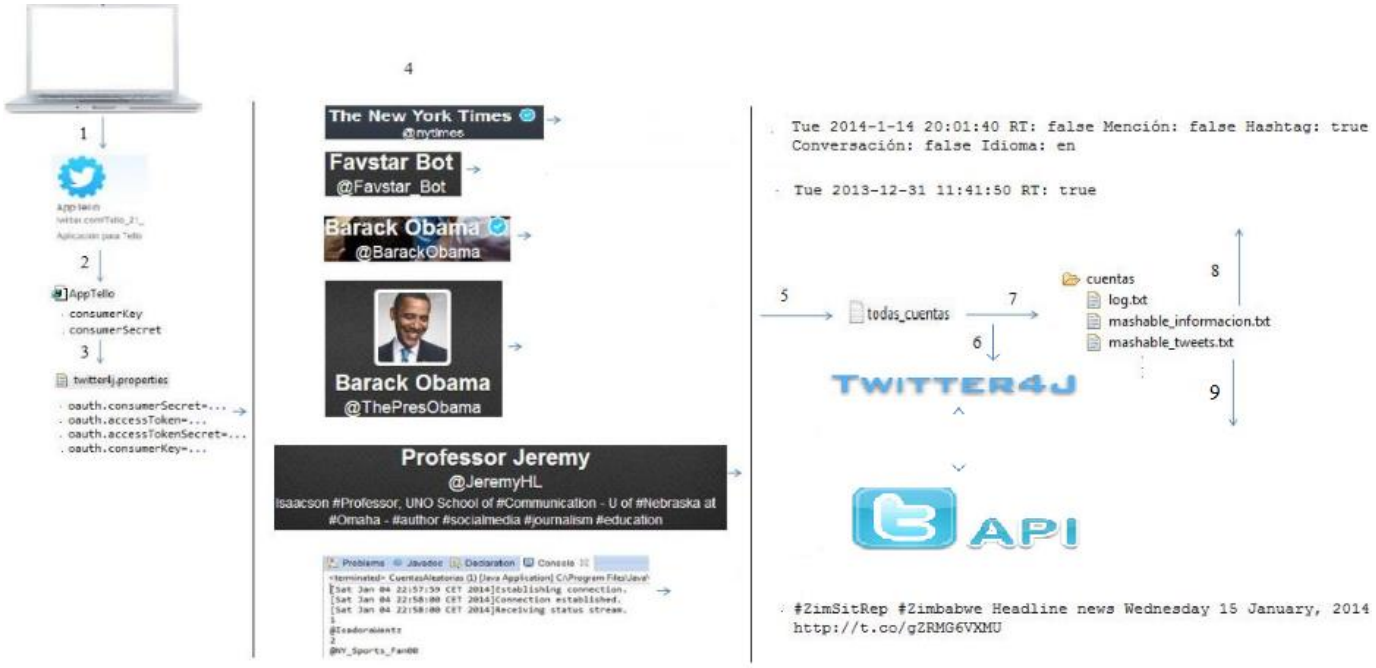


Figura 39: Diagrama general de la aplicación (1)

Módulo 4:

10. **Detección del idioma del tweet**. Para una mayor fiabilidad, detectamos el idioma de dos formas posibles: la primera es mediante el Proyecto Language-detection de Google code, y la otra es mediante la librería de Java para la API de Twitter llamada Twitter4j. Nos interesa cuando ambos resultados coinciden con "en", que significa que el tweet está escrito en inglés.

Módulo 5:

11. **Análisis lingüístico.** Calculamos una serie de indicadores para detectar la calidad de un usuario en Twitter, analizando la sintaxis de los tweets y la variedad de palabras utilizadas.
12. **Parser de Stanford.** Usamos el parser de Stanford para analizar sintácticamente cada tweet, y determinar la calidad lingüística en función de la profundidad del árbol sintáctico, el número de patrones sintácticos distintos en tweets, si contiene oraciones subordinadas, sintagmas adjetivales y adverbiales.

13. **Wiktionary.** Utilizamos la lista de palabras en inglés que nos proporciona el Wiktionary para conocer la variedad de palabras que aparecen en los tweets, centrándonos en las últimas 20.000 palabras ya que son las menos usadas, y por tanto, más cultas.
14. **Análisis general.** Hacemos uso del fichero generado en el punto 9, donde aparece información específica de cada tweet.
15. Gracias al fichero de información, podemos calcular los indicadores siguientes: porcentaje de tweets que son RT, conversación, y los que contienen hashtags, menciones, al igual que el número de seguidores y gente a la que siguen.
16. Para el caso de los enlaces, los detectamos dividiendo el tweet en tokens, y comprobando si los cuatro primeros caracteres son "http".
17. **Análisis temporal**. Calculamos la frecuencia de twitteo analizando el tiempo transcurrido en segundos entre un tweet y su anterior.
18. **Valor máximo** de todas las diferencias de segundos obtenidas.
19. **Desviación típica** de las diferencias de segundos entre un tweet y su anterior.

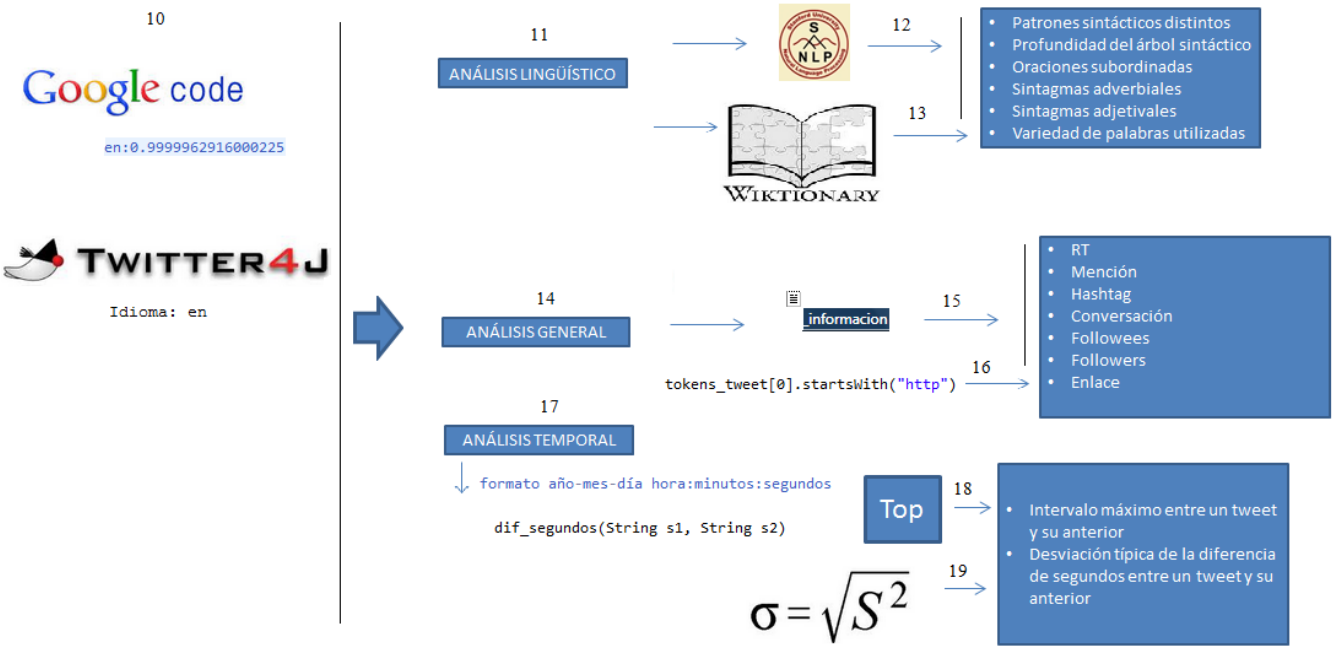


Figura 40: Diagrama general de la aplicación (2)

Módulo 6:

20. **Generamos un fichero para cada identificador**. Dicho fichero consta de 6 columnas, donde cada columna contiene los resultados del tipo de cuenta en cuestión: x1 (cuentas de periódicos), x2(cuentas bots), x3(cuentas fake), x4(cuentas simuladas), x5(cuentas de calidad), x6(cuentas aleatorias) donde cada columna contiene los resultados de las 40 cuentas del tipo de cuenta correspondiente.
21. **Script.** Haciendo uso del paquete R, hemos creado un script que para ejecutarlo necesitamos el fichero para cada identificador del punto20, y poner un nombre para el fichero que queremos generar.
22. **Box-plots**. Tras la ejecución del script, se ha generado el gráfico que contiene 6 box-plots con la representación de los valores. Ejecutamos el script tantas veces como identificadores se necesiten.

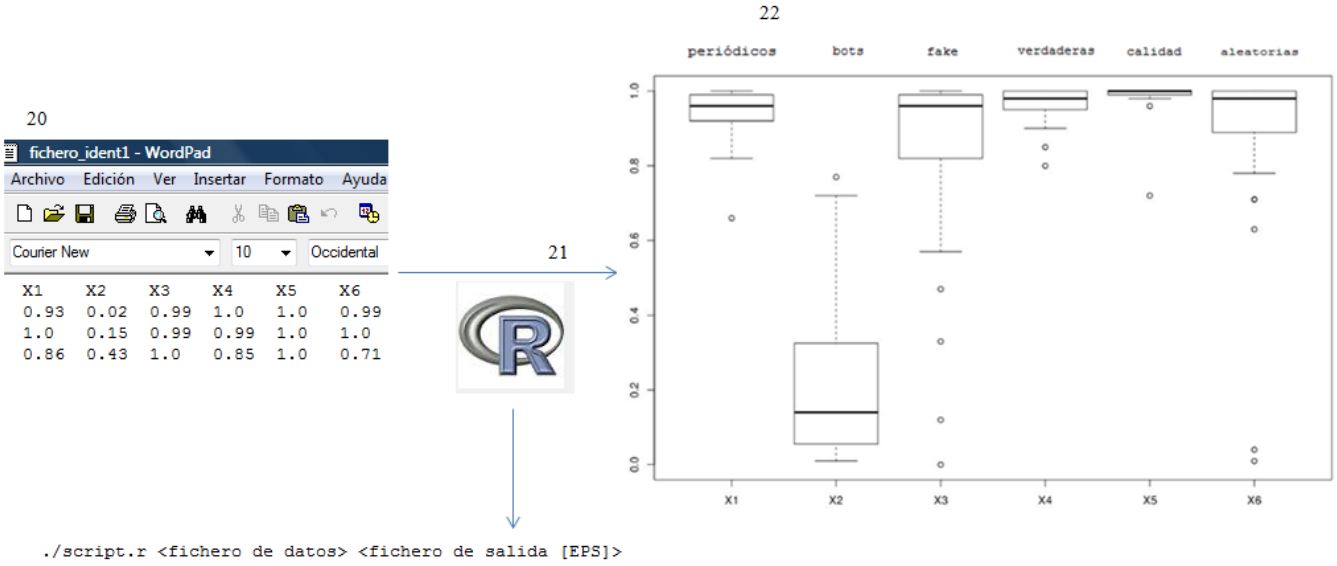


Figura 41: Diagrama general de la aplicación (3)

4.2. Requisitos funcionales

El proyecto tiene como propósito disponer de una herramienta que se conecte con la red social Twitter, a través del protocolo de autenticación OAuth, y permita analizar los tweets de los usuarios elegidos para su posible diferenciación.

A pesar de que el lenguaje de programación usado para el desarrollo del trabajo de investigación es Java: el parser de Stanford analiza los textos en inglés, la librería para el API de Twitter es de Java también y la librería de Java para el detector de idiomas, se podría realizar en otros lenguajes sin problemas.

Siempre que los usuarios no tengan protegidas sus cuentas en Twitter, se podrá hacer una búsqueda libre de usuarios a analizar, y obtener datos de la cuenta que algunas han autorizado (ubicación de los tweets) y otros son públicos por defecto pero sin comprometer a nadie (nombre, id). Además, se debe cumplir en todo momento con las restricciones impuestas por la API de Twitter que se han detallado en el apartado 2.4.3.

4.3. Requisitos no funcionales

El sistema debe tener acceso a las APIs de la red social Twitter. En cuanto a las peticiones externas que se realice con la API de Twitter, se debe cumplir la restricción de transferencia de datos protegidos entre el usuario y proveedores de servicio a través de aplicaciones, utilizando el protocolo OAuth y cumpliendo con la Ley Orgánica de Protección de Datos.
La plataforma debe ser de fácil manejo para el usuario y con el código reutilizable para una posible actualización o ampliación futura, con una buena distinción de sus módulos y componentes, de manera que se pueda hacer uso de cada uno de ellos por separado sin afectar la funcionalidad del resto.
Dado el bajo coste de los materiales empleados, la herramienta está disponible para todo el público, que podrán utilizarla para cualquier uso y sacarla todo el rendimiento posible.

4.4. Diseño

En este apartado se muestra todo lo relacionado con el diseño de la aplicación. Inicialmente aparecen las alternativas de diseño en cuanto a la arquitectura, y posteriormente se muestra una imagen dedicada a la visualización de los resultados obtenidos.

4.4.1. Alternativas de diseño

Explicamos las diferentes alternativas de diseño que se pensaron al comienzo del proyecto, comparando cada una de ellas, mostrando unos diagramas conceptuales, y explicando la elección final de diseño.

4.4.1.1. Alternativa de diseño 1

Esta alternativa consiste en guardar los tweets en un fichero de texto, para luego leerlos y procesarlos.

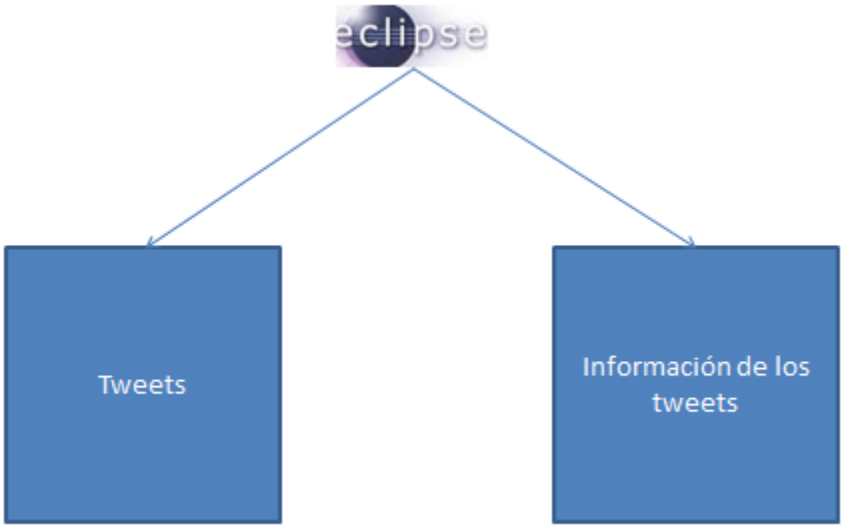


Figura 42: Alternativa de diseño 1

En este caso se almacenan los tweets en ficheros de texto, junto al código fuente. Hay mayor cantidad de datos en el directorio de la aplicación, pero es una forma muy útil de acceder a ellos.

4.4.1.2. Alternativa de diseño 2

En este caso hemos pensado guardar los tweets en una base de datos mysql, por lo que toda la información ya no se encontraría en el código, y además los datos estarían más organizados.

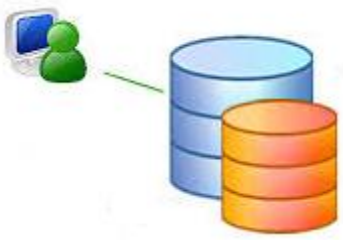


Figura 43: Alternativa de diseño 2

4.4.1.3. Alternativa de diseño 3

Esta última alternativa consiste en guardar los tweets en una base de datos y que resida en un servidor externo.

Figura 44: Alternativa de diseño 3

En este caso se ha pensado guardar todos los tweets en una base de datos al igual que en la alternativa anterior, pero de esta forma residirían en un servidor externo. Ocuparía menos espacios y habría que hacer uso de internet.

4.4.1.4. Decisión final en cuanto al diseño

Finalmente se decidió por la alternativa 1, debido a la comodidad de tener los tweets junto a todo el proyecto, además de una mayor independencia de los datos al utilizar ficheros, y una mayor eficiencia en la recogida de los tweets. Con la alternativa 2 tendríamos que insertar cada tweet en la base de datos, de modo que sería más laborioso para un tema que no requiere tanta complejidad. La alternativa 3 la descartamos porque requiere una complejidad que a nuestro modo de pensar es

innecesaria para este caso, se pensaron todas en su momento pero llegamos a la conclusión de utilizar ficheros como forma de almacenamiento de los datos.

4.4.2. Directorios y ficheros

- **Ficheros de los tweets e información**

Hemos decidido generar un fichero para los tweets de cada usuario, donde aparece un tweet por línea. Además, otro fichero por cada usuario que contiene la información relativa a ese tweet en cuestión, de forma que podemos diferenciar cada parte por separado y se adapta a cualquier necesidad del cliente, y no exclusivamente para el objetivo concreto de este proyecto en cuestión. En nuestro caso, incluimos la fecha del tweet, y los valores true o false dependiendo si el tweet es RT, conversación, y si contiene hashtags y menciones. También el idioma del mismo.

❖ **Fichero todas_cuentas**

En cuanto al fichero que contiene los nombres de los usuarios que queremos estudiar, hemos decidido poner un usuario por cada línea, e ir analizando de 40 en 40 ya que están ordenados por tipos que es lo que nos interesa.

❖ **Fichero Wiktionary**

En cuanto al diseño de este fichero, tenemos por cada línea, la palabra en inglés ordenada por su frecuencia de uso y también aparece la frecuencia, aunque nosotros solo utilizamos la posición de la palabra para conocer si es de uso corriente o no, la frecuencia puede servir para futuras investigaciones más detalladas.

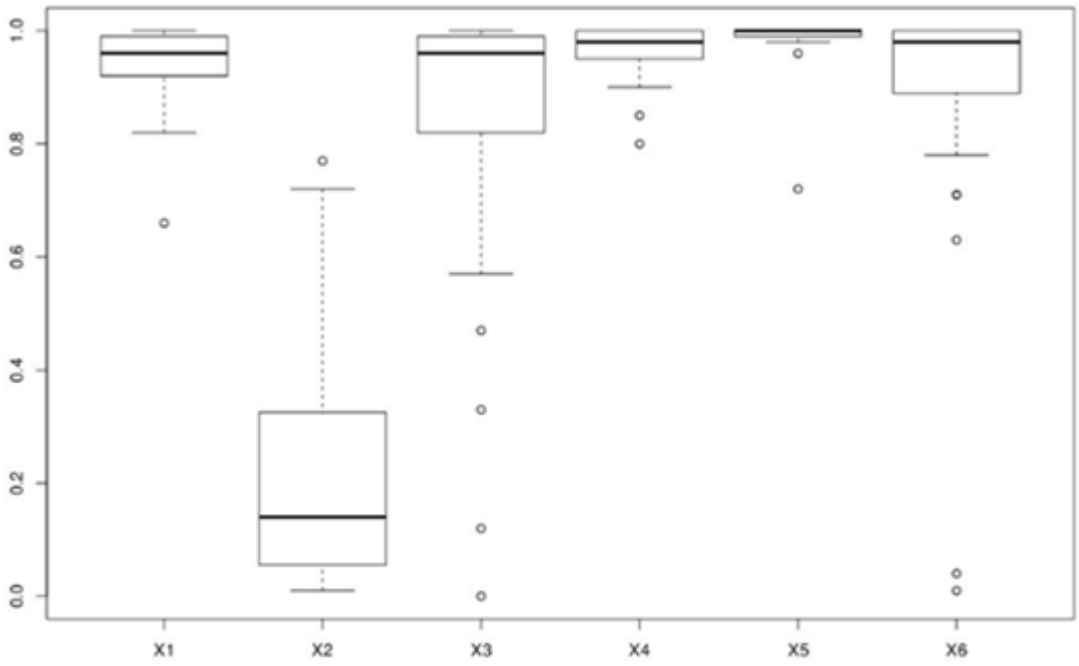
❖ **Ficheros generados como resultado de la ejecución del programa**

Generamos un fichero a modo de resumen con la información de cada identificador calculado por cada usuario. Aparece el número del identificador, descripción breve y el resultado obtenido, lo que nos permite acceder en un momento dado a la información específica que nos pueda interesar sobre cada usuario en particular, ya que agrupamos los usuarios en diferentes tipos y muchas veces se obtienen valores atípicos que se requiere una explicación.

También generamos un fichero por cada identificador con 240 resultados referentes a cada usuario, que posteriormente usaremos para generar su representación gráfica y poder visualizar los box-plots correspondientes.

4.4.3. Visualización de los resultados

Decidimos visualizar el conjunto de los resultados usando los diagramas de cajas conocidos como box-plots, ya que es una buena forma de observar la distribución de los datos y poder diferenciar diferentes tipos de cuentas. Decidimos los box-plots porque generamos una cantidad de valores considerable, y necesitamos alguna herramienta de estadística descriptiva para poder comparar todos los valores, y además, la forma visual es la mejor manera de demostrar el objetivo de nuestro trabajo de investigación.



4.5. Descripción de los módulos o rutinas principales

Se hace una descripción de las rutinas principales que consta el proyecto, diferenciando cada módulo con notoriedad.

4.5.1. Módulo 1: Registro de una app en Twittter

Este módulo se centra en explicar la posibilidad de desarrollar nuestra propia aplicación para que se conecte y obtenga información de Twitter, sin necesidad de acceder a la página web.

En primer lugar, tenemos que registrar nuestra aplicación para tener acceso a la cuenta de Twitter, siendo @Tello_21_ nuestra cuenta. Realizamos este proceso con el protocolo OAuth, el cual permite al usuario autorizar la aplicación para actuar en su nombre sin compartir la contraseña de Twitter. El lugar para acceder al registro se hace desde [24].

Figura 45: Registro de una App en Twitter

El formulario de registro consta de los siguientes campos:

- **Icono de la aplicación:** opción de subir el icono de la aplicación que se desee, o con la imagen por defecto que dispone Twitter.
- **Nombre:** indicamos el nombre de la aplicación. Nosotros elegimos "AppTello".
- **Descripción:** añadir una breve descripción de las características y uso de la app.

[24] https://twitter.com/oauth_clients/new

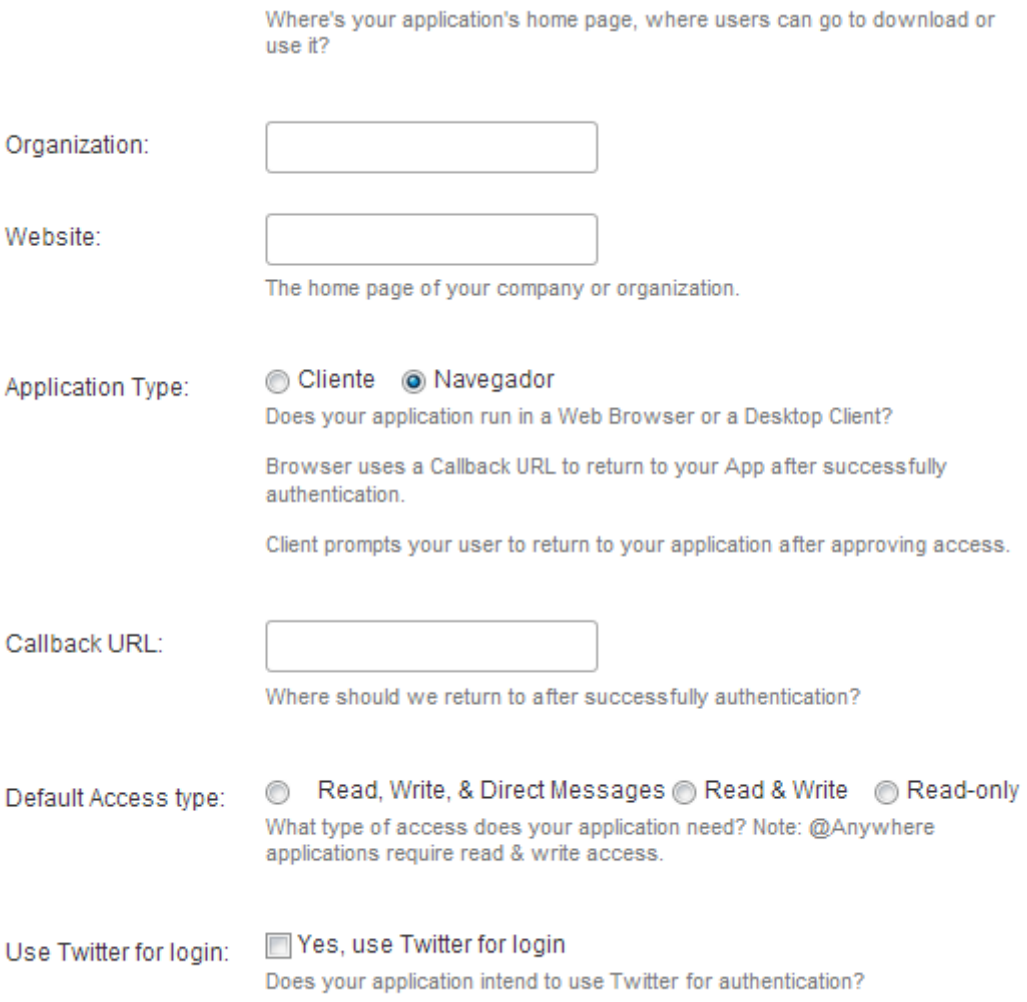


Figura 46: Últimos campos del registro de una App en Twitter

Otro campo es referente a la **web** de una compañía y otro a la **organización**. Si deseamos utilizar Twitter para autenticarnos, entonces la autenticación satisfactoria de lo escrito en el campo "Callback URL" será invocada.

Dada la finalidad que queramos dar a la información obtenida con la aplicación, podemos elegir diferentes tipos de acceso:

- **Lectura, escritura y mensajes directos:** los permisos que dispone la aplicación le permiten leer y añadir contenido nuevo, al igual que acceder a los mensajes privados del usuario.
- **Lectura y escritura:** se puede leer el contenido de Twitter y añadir nuevos tweets.

- **Solo lectura:** se da permisos únicamente de lectura para el acceso al contenido de Twitter. Es la opción que hemos elegido ya que solo nos interesan los tweets de los usuarios, así como algunas características más, sin añadir o modificar nada.

Nos logueamos con la cuenta de Twitter que hemos creado (@Tello_21_ en mi caso). Posteriormente, aparece una serie de información relativa al formulario rellenado, y otras nuevas, como es el caso de "consumer key" y "consumer secret". Estos datos nos interesan ya que al inicio, la App no tiene permisos para acceder a la cuenta del usuario, y es por ello la necesidad de adquirir el token de acceso. Creamos un fichero "AppTello.key" con extensión .key donde insertamos en la primera línea el "consumer key", y en la segunda línea "consumer secret", que son las claves del registro. El motivo de guardar esta información en un fichero es porque solo se realiza una vez, por lo que será mejor reutilizar dicha información tantas veces como se necesite, y no generarlo de nuevo cada vez.

Tenemos una clase para generar el token de acceso, necesario para la autentificación por Oauth, donde validaremos el acceso de la aplicación a la cuenta de Twitter asociada. Primeramente leemos del fichero "AppTello.key", donde se encuentran las dos claves que nos otorga Twitter al registrar nuestra aplicación. Seleccionamos autorizar aplicación para acceder a mi aplicación AppTello, pulsando el botón azul en la siguiente imagen:

Figura 47: Abriendo el navegador autorizar la aplicación

Comprobamos que el acceso está autorizado por el usuario, copiando el pin que se muestra en la Figura 49 e introduciéndolo en nuestro programa.

Figura 48: Generando el pin para completar el proceso de autorización

Finalmente, se genera el fichero twitter4j.properties con la siguiente información:

Contenido del fichero twitter4j.properties:

-Nombre

-Fecha

Consumer key y consumer secret (claves de la aplicación) y acces token y access token secret (tokens de acceso para acceder a Twitter).

El contenido del fichero Twitter4j.properties se muestra en la imagen siguiente:

```
#twitter4j.properties
#Sat Jan 04 14:20:32 CET 2014
oauth.consumerSecret=DhVRKOQKT08ppC002bkFoip81MmJbY53ZaCISpJ7Q
oauth.accessToken=407332434-pmX8BBc8g3MBmBArkxuyrv5ukVROAXEhckOdrU6L
oauth.accessTokenSecret=xLpDfrUaZXjj6YcsGwOfFdgze7CawhAuuNRh5TpBAyGZG
oauth.consumerKey=YXdWjwDEEDcTRqssxK2EQ
```

Figura 49: Contenido del fichero Twitter4j.properties

Una vez generado el token de acceso, el usuario puede pedir recursos al servidor por estar autorizado, siempre y cuando se pase en cada petición HTTP la credencial de token que ha autenticado. Estos recursos pueden ser los tweets actuales o antiguos, nombres de cuenta de twitter que twittean en tiempo real, seguidores y personas a las que se sigue.

4.5.2. Módulo 2: Búsqueda de cuentas de Twitter

En primer lugar, tuvimos que buscar cuentas de Twitter para los seis tipos de clases que hemos estudiado a lo largo del experimento, con el requisito de que escribieran en inglés, que es el idioma que estamos analizando por ser el universal, y porque utilizamos un analizador sintáctico para el análisis de textos en inglés. En total se buscaron 240 cuentas, con 40 cuentas para cada grupo tal y como se comenta en el Resumen del proyecto. Dichas cuentas las insertamos en un fichero de texto llamado "todas_cuentas.txt", para su posterior lectura de fichero de 40 en 40, ordenadas de la siguiente manera:

- Las 40 primeras cuentas son de medios de comunicación, todas ellas están verificadas oficialmente por Twitter y las obtuvimos en [26].
- Cuentas bots reconocidas por páginas con buen criterio y justificación [30], y otras escogidas mediante una búsqueda exhaustiva por Twitter, donde claramente se pueden apreciar que no son cuentas escritas por humanos y sí son utilizadas por programas automatizados. Excluimos cuentas bots que son conversacionales y también aquellas que se dedican a tuitear los titulares de periódicos o a retuitear únicamente. Nuestro estudio se centra en comprobar la calidad de estos usuarios, y en base a los resultados poder diferenciar unas cuentas de otras, y en particular cuentas bots de humanos.
- Cuentas simuladas sacadas en [27], [28], [29] y algunas de forma manual, y corroborándolas posteriormente por nosotros. Hemos obtenido estas cuentas (llamadas fake en los gráficos generados) referentes a las cuentas que parodian o imitan a las que hemos llamado verdaderas, de modo que ha sido fácil la búsqueda en cuanto a las celebridades famosas y más complicado sus cuentas falsas.
- 40 cuentas que llamamos "verdaderas" ya que son las parodiadas o imitadas por las cuentas fake. Suelen ser personas famosas por lo que están verificadas la mayoría para evitar precisamente que las imiten o parodien.
- Las siguientes 40 cuentas son de investigadores y profesores de Universidad, periodistas, muchas de ellas verificadas por ser personalidades reconocidas en su entorno y fuera de él, y las hemos llamado de calidad. La búsqueda la realizamos manualmente en Twitter observando la lista de seguidores de cuentas verificadas de Universidades prestigiosas del mundo como Stanford y Cambridge.

- Cuentas aleatorias, donde nos conectamos a Twitter y sacamos las primeras 40 cuentas a las que pertenecen los primeros tweets leídos. De esta forma obtenemos cuentas normales donde podemos obtener todo tipo de cuentas.

4.5.3. Módulo 3: Generación de tweets

Este módulo es específico para la obtención de los tweets que queremos analizar. La mayoría de ellos los hemos generado con la clase GenerarTweets.java, en concreto 200 cuentas de usuarios, y las 40 restantes las hemos sacado con la clase CuentasAleatorias.java obtenidas mediante el Streaming API.

4.5.3.1. Usuarios ya escogidos

La clase GenerarTweets.java se encarga de generar un fichero con la cantidad de tweets que nosotros queramos, en nuestro caso decidimos que fueron 1400 tweets. También genera un fichero con la información general en Twitter: fecha en que se posteó, si es RT, hashtag, enlace, conversación y su idioma. Debido al límite impuesto por Twitter, no podemos obtener los tweets de los 240 usuarios a la vez, sino que hay que ir haciéndolo cada cierto tiempo, de 10 en 10 usuarios y en intervalos de 15 minutos, debido al límite del API en la versión 1.1.

En primer lugar, introducimos a mano, en un fichero de texto los nombres de los usuarios en Twitter que queremos analizar y, usando la clase anterior leemos todos ellos con el método leer_cuentas () y rellenamos un array de string para poder analizarlos. Creamos el directorio cuentas donde se almacenarán los dos ficheros que creamos con esta clase. Después, conseguimos una instancia de Twitter con credenciales por defecto, invocamos al método lookupUsers () pasándole por parámetro el array con los nombres de las cuentas, y ya tenemos la lista de usuarios de Twitter.

La recorremos y por cada usuario genero dos ficheros dentro del directorio cuentas: nombre_usuario_tweets.txt para guardar los tweets de cada usuario y nombre_usuario_information.txt para guardar información de cada tweet de cada usuario. La forma de obtener los tweets es por paginación, de forma tenemos que indicar la página y el número de elementos por página que queremos obtener del timeline del usuario.

Por cada usuario, por cada estado obtenemos la información que nos interesa. Hemos usado dos formatos en el fichero de información para analizar algunas características de la cuenta de un usuario en Twitter:

- En el caso de que el tweet sea RT, solo se tendrá información de la fecha en que se posteó, y de un valor a true para conocer que dicho tweet es RT, ya que esta información nos sirve para contabilizar el número de RTs y, por tanto, poder calcular el porcentaje de ellos. La fecha la tenemos que tener en todos los tweets para calcular los indicadores temporales. Para conocer cualquiera de los identificadores restantes no nos valen, porque no están escritos por el propio usuario.

  Formato de ejemplo: `Mon 2014-1-13 22:32:40 RT: true`
- Para el otro caso en el que el valor de RT sea falso, obtenemos información que nos resulta útil para calcular algunos de los identificadores generales como: si el tweet contiene mención, hashtags o si es conversación. Con respecto al idioma, obtenemos el idioma del tweet de acuerdo a la información que nos da el API de Twitter, aunque luego la tendremos que comprobar con el idioma obteniedo con el proyecto de Google, y quedarnos con la salida "en" del detector de idiomas.

  Formato de ejemplo: `Mon 2014-1-13 22:26:17 RT: false Mención: false Hashtag: false Conversación: false Idioma: en`

Los detalles más característicos los detallamos en el diagrama de flujo a continuación:

Diagrama de flujo de la clase GenerarTweets.java:

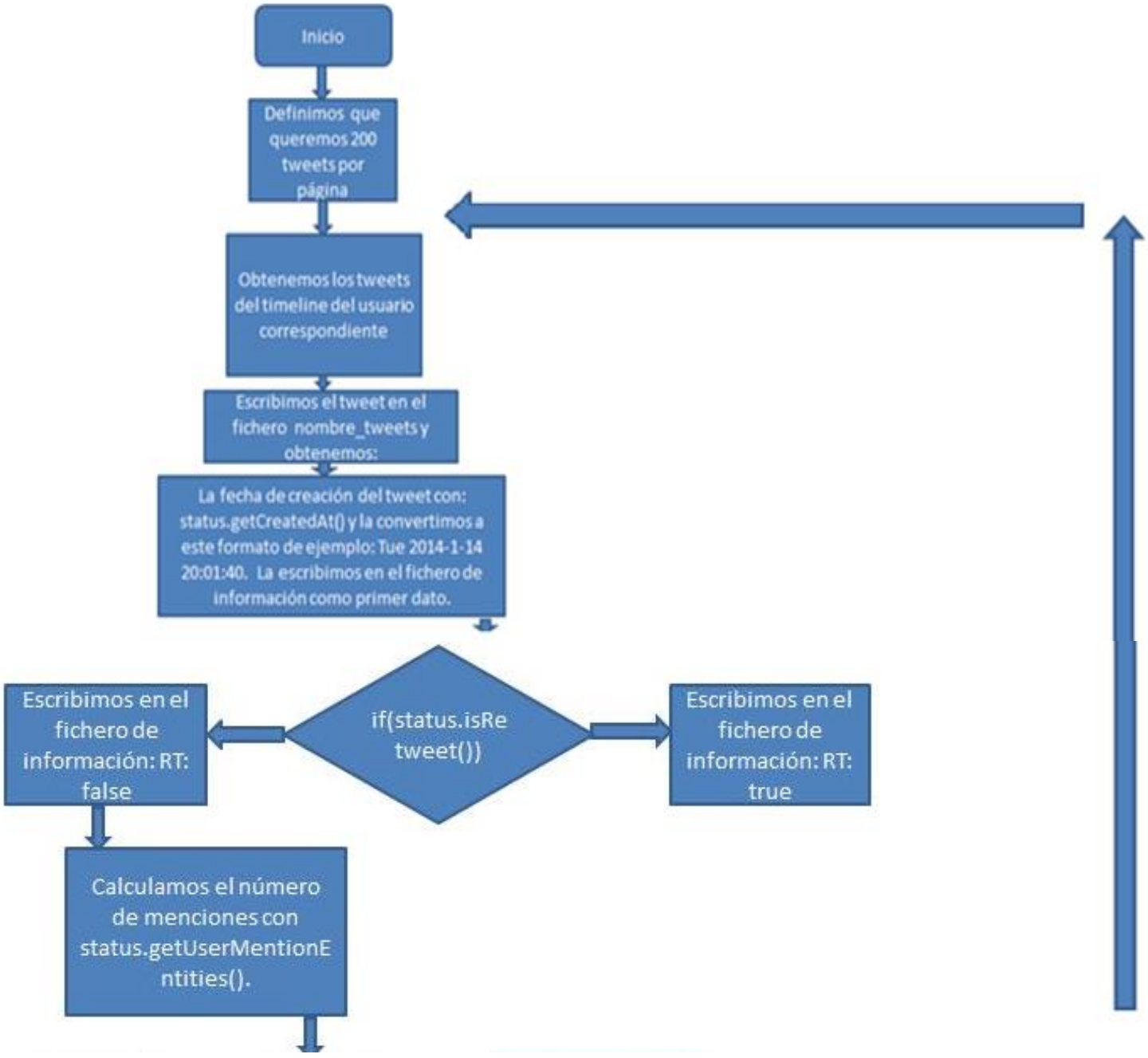


Figura 50: Diagrama de flujo de la clase Generartweets.java (parte 1)

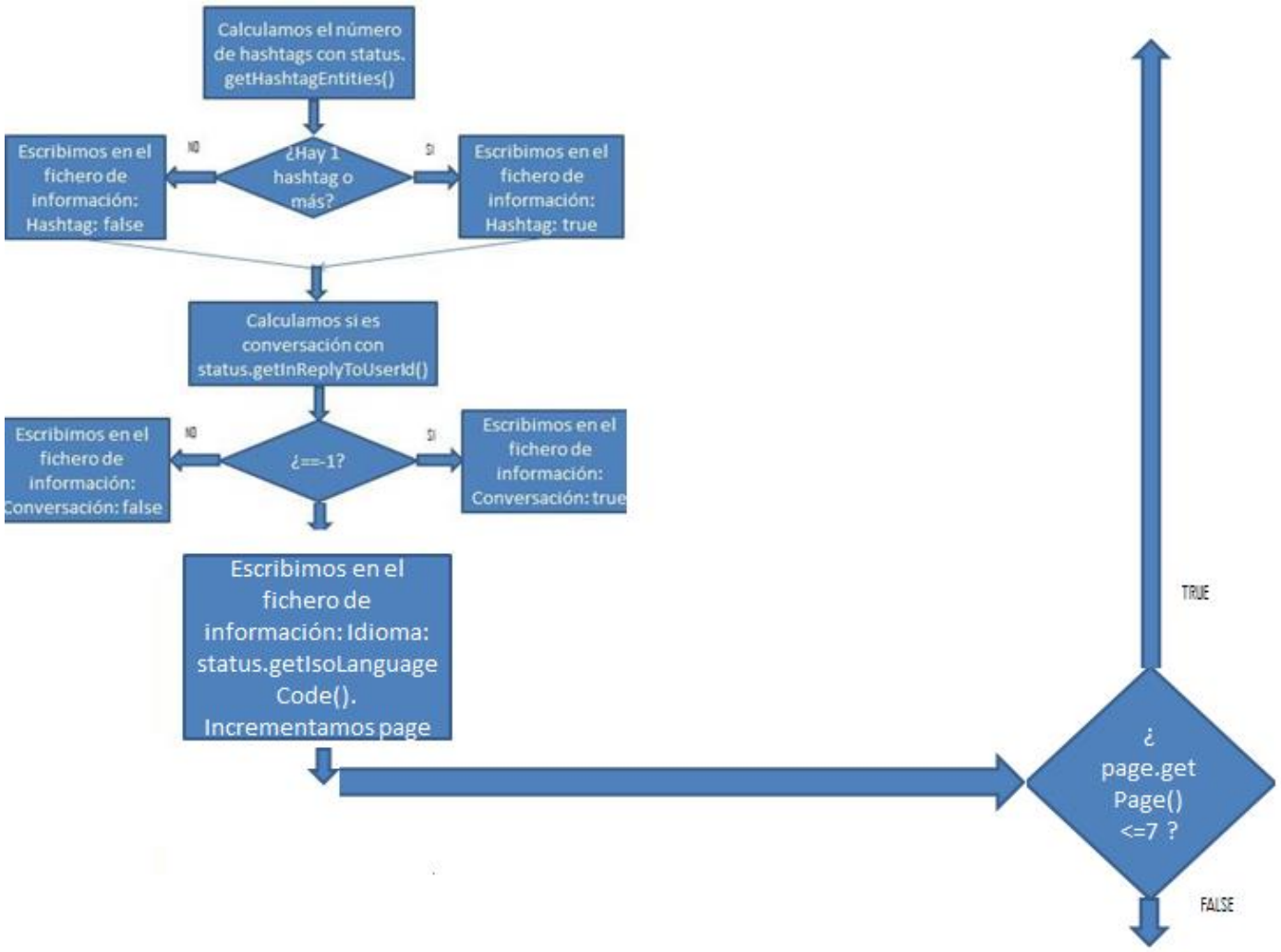

Calculamos el número de hashtags con status. getHashtagEntities()
¿Hay 1 hashtag o más?
NO
SI
Escribimos en el fichero de información: Hashtag: false
Escribimos en el fichero de información: Hashtag: true
Calculamos si es conversación con status.getInReplyToUserId()
¿==-1?
NO
SI
Escribimos en el fichero de información: Conversación: false
Escribimos en el fichero de información: Conversación: true
Escribimos en el fichero de información: Idioma: status.getIsoLanguage Code(). Incrementamos page
¿ page.get Page() <=7 ?
TRUE
FALSE

Figura 51: Diagrama de flujo de la clase Generartweets.java (partes 2 y 3)

4.5.3.2. Usuarios aleatorios

Para obtener las 40 cuentas aleatorias, utilizamos el **Streaming API.** Nuestra clase "CuentasAleatorias.java" implementa StatusListener para obtener los tweets necesarios en tiempo real.

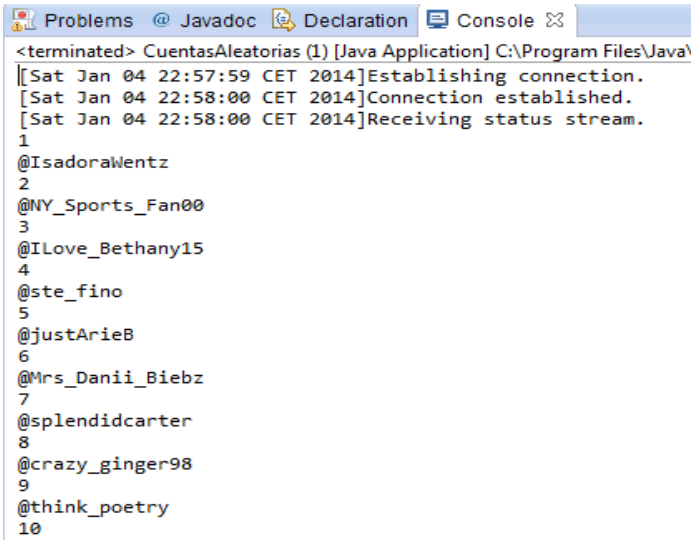


Figura 52: Obtención de usuarios usando el Streaming API

Para detenerlo, utilizo twitterStream.shutdown (). El método shutdown () cierra el hilo distribuidor interno compartido por todas las instancias TwitterStream. Como solo quiero 40 cuentas aleatorias, cuando el contador llega a 40 se detiene:

Figura 53: Cerrando el hilo distribuidor interno compartido por todas las instancias TwitterStream

Para las cuentas normales, la idea es conectarse en cualquier momento a Twitter y coger las 40 primeras cuentas a las que pertenezcan los primeros tweets leídos, con la condición de que estén en inglés y que tengan al menos 10 tweets desde que se registraron en la cuenta de Twitter (10 tweets mínimo para así poder tener algunos datos que analizar).

4.5.4. Módulo 4: Detectar el idioma del tweet

Para una mayor precisión hemos decidido detectar el idioma de dos formas posibles: la primera es mediante el Proyecto Language-detection de Google code, y la otra es mediante la librería de Java para la API de Twitter llamada Twitter4j.

4.5.4.1. Proyecto de Google

El procedimiento llevado a cabo para detectar el idioma de los tweets usando el proyecto de Google se explica a continuación:

Inicializar biblioteca

Antes de detectar el idioma del tweet, creamos el método init y llamamos al método loadProfile de la clase DetectorFactory, para cargar los perfiles de idiomas del directorio pasado como parámetro. Los perfiles lingüísticos se incluyen en esta biblioteca con el directorio profiles, y lo guardamos dentro del directorio principal del Proyecto. Como nuestro análisis se centra en el idioma inglés, vamos a centrarnos básicamente en el perfil "en".

Establecer texto de destino

La clase Detector es una interfaz de esta biblioteca de detección de idiomas. Esta clase puede ser sólo instanciada través de la clase DetectorFactory. Detector sirve para detectar el idioma del texto especificado, usando el método append().

append () tiene dos versiones de parámetro. Uno recibe Reader para leer el texto del lector de entrada especificado. Si el tamaño total del texto de destino excede del tamaño límite especificado saltará una excepción. El otro parámetro posible recibe un String y lo que hace es anexar el texto destino para la detección del idioma, que es el que utilizamos, pasándole el tweet en cuestión. En la siguiente captura de pantalla podemos observar el método implementado donde se le llama pasándole el tweet, y obtenemos el perfil del idioma de ese tweet. Por último, nos queda comprobar que ese perfil pertenece al idioma inglés, es decir, si el String devuelto es: "en".

```
public String detect(String text) throws LangDetectException {
    Detector detector = DetectorFactory.create();
    detector.append(text);
    return detector.detect();
}
```

Figura 54: Método para detectar el idioma de un tweet usando el Proyecto de Google

Si se desea detectar nuevamente el idioma, habrá que crear una nueva instancia del detector.

4.5.4.2. Usando la librería Twitter4j

Cuando generamos el fichero que contiene información del usuario en Twitter, en el módulo GenerarTweets, una de las peticiones que hacemos a Twitter es el idioma en que se ha escrito el tweet, gracias al método getIsoLanguageCode(). Este método devuelve el perfil del idioma. Por lo tanto, simplemente tenemos que leer la línea del fichero (hay una línea de información por cada tweet, y concuerda el número de línea con el número de tweet), dividirlo en tokens y comprobar que el token en la posición 12 es igual a "en".

Una vez usado ambos sistemas de detección del idioma del tweet, lo único que nos falta es comprobar que ambos coinciden en que el perfil del idioma es "en". De esta forma tenemos más fiabilidad de que el tweet está escrito en inglés.

4.5.5. Módulo 5: Indicadores

Para el desarrollo del Trabajo Fin de Grado, hemos analizado una serie de indicadores que nos pudieran servir para detectar la calidad de un usuario en Twitter. La mayoría son lingüísticos, pero también se encuentran indicadores generales referentes a las características de la cuenta de un usuario en Twitter, y algunos temporales en relación a la frecuencia de twitteo. Aquí comentamos cómo se han obtenido dichos identificadores, en otro apartado más adelante se analizarán los resultados obtenidos, sacando nuestras propias hipótesis y, por último, las conclusiones obtenidas.

4.5.5.1. Indicadores lingüísticos

Los utilizamos para comprobar la calidad de un usuario en Twitter desde el punto de vista de la sintaxis utilizada en los tweets posteados. Algunos se centran en el análisis sintáctico, y otros en las palabras usadas para formular los tweets:

4.5.5.1.1. Análisis sintáctico

Los indicadores están basados en el análisis sintáctico de los tweets escritos por el usuario de Twitter. Hemos analizado los siguientes indicadores en esta categoría:

Indicador 1. Número de patrones sintácticos distintos en tweets dividido por el número de tweets

Por patrón nos referimos al árbol sintáctico que generamos con el parser de la Universidad de Stanford quitándole las palabras del texto del tweet. Este indicador nos interesa porque en las cuentas bots el número de patrones sintácticos distintos en los tweets que no son RT y, por tanto, generados por el programa informático deben ser reducidos, ya que van a utilizar generalmente el mismo patrón. No tenemos en cuenta los tweets que son RT ya que son escritos por humanos y deben tener una variabilidad muy elevada (a no ser que hagan RT a otra cuenta bot, para evitar errores los RT no los tenemos en cuenta para el patrón y sólo lo usamos para calcular el porcentaje de tweets que son RT).

Primeramente utilizamos el parser de Stanford para analizar el tweet y obtenemos el árbol sintáctico. Luego lo descomponemos en un array de árboles, y por cada uno de ellos obtenemos el etiquetado sintáctico, y lo convertimos a tipo String para así meterlo en el vector general, el cual va

a contener las etiquetas y las palabras. Seguidamente rellenamos otro vector con las palabras que componen el tweet. Ahora tenemos que recorrer el vector de palabras, y por cada palabra recorro el vector que contiene tanto el etiquetado como las propias palabras, y cuando coincidan las borro del vector general, porque solo me interesa el etiquetado sintáctico para el patrón sintáctico. En el caso de que el tweet contenga enlaces, puede ocurrir que el enlace se encuentre al inicio del tweet, por lo que se comprueba si el flag enlace_principio está a true, si es así, concatenamos "enlace" al inicio del patrón. En el caso de que haya un enlace al final del tweet, se añadirá "patrón" al final. Si el enlace se encuentra en el medio del tweet, directamente lo borramos porque no queremos obtener el etiquetado del enlace ya que no hay un etiquetado específico para ello.

Nos interesa este identificador para conocer el número de patrones sintácticos diferentes, de forma que vamos comparando unos patrones con otros y almacenamos los patrones distintos. Para ello, los introducimos en una HashMap para que sea escalable. Para cada tweet que esté en inglés y no sea RT comprobamos en la HashMap si ya se encuentra. En el caso de que el patrón no esté, lo introducimos con su clave y valor, de modo que la HashMap está declarada con una clave de tipo Integer y un patrón de tipo String, de forma que la clave será un número entero y el valor será el patrón sintáctico del tweet obtenido en el programa. Al final, si queremos conocer el número de patrones sintáctico distintos, es tan sencillo como calcular el tamaño de la HashMap con el método size (). La clave_patrón es el número de patrones sintácticos distintos, al inicio tenemos que meter el primer valor (clave_patron es 0) para poder comparar la segunda vez.

Diagrama de flujo del **patrón sintáctico**:

Figura 55: Diagrama de flujo del patrón sintáctico (Parte 1)

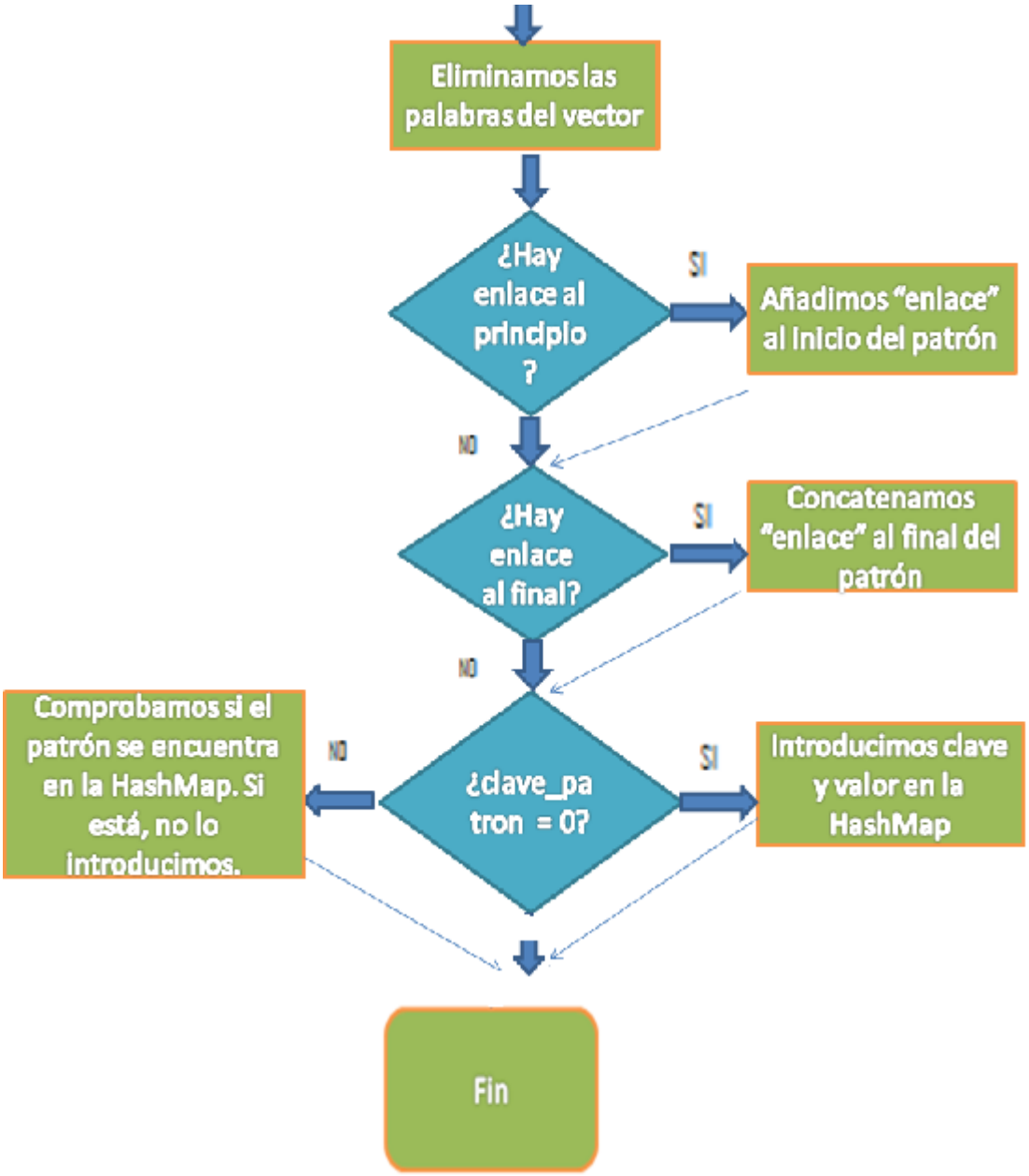


Figura 56: Diagrama de flujo del patrón sintáctico (2)

Ejemplo de cuenta bot en Twitter utilizando el mismo patrón sintáctico:

Figura 57: Ejemplo de cuenta bot en twitter usando el mismo patrón sintáctico

Se aprecia en la imagen que los tres tweets siguen un patrón igual. De hecho, los dos primeros se diferencian únicamente en el enlace, y el tercero con los otros dos en el número 250 en vez de 50, además del enlace. Se observa que se trata de una cuenta bot que escribe a usuarios aleatorios dándoles información sobre el número de tweets favoritos que tiene.

Indicador 2. Desviación típica de la profundidad del árbol sintáctico de cada tweet

Con profundidad nos referimos a la distancia desde el nodo raíz (ROOT) hasta el último nodo del árbol sintáctico que genera el parser. Por ejemplo:

Frase: Sales executives were examining the figures with great care.

Árbol sintáctico que genera el parser:

(ROOT

(S

(NP (NNS Sales) (NNS executives))

(VP (VBD were)

(VP (VBG examining)

(NP (DT the) (NNS figures))

(PP (IN with)

(NP (JJ great) (NN care)))))))

Lo ponemos de esta forma para mayor claridad visual:

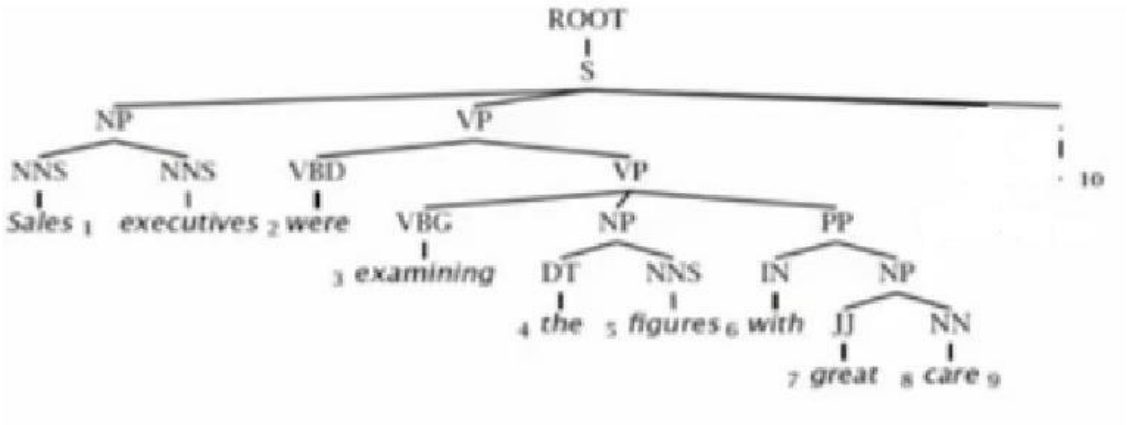


Figura 58: Árbol sintáctico de una frase de ejemplo

Para el cálculo de la profundidad, la función depth() de la clase Tree lo que hace es empezar por el nodo raíz que es ROOT, baja hasta S y ya es un paso, luego tiene tres caminos posibles: NP VP i, debe ir por el camino que tenga más nodos hacia abajo, por tanto va a VP y ya son dos pasos. Baja hasta VP, luego a PP, NP, JJ-great o NN-care (da igual porque ya estamos al final del árbol). Por lo que la **profundidad** de este caso concreto es **7**, el camino seguido sería: ROOT->S->VP->VP->PP->NP->((JJ->great) | (NN->care)).

Ese paso sería para el cálculo de la profundidad de un tweet, pero nos interesa la media de la profundidad de los 1000 tweets que estén en inglés y que no sean RT, para poder obtener su desviación típica. Por tanto, calculamos la profundidad de cada tweet y la insertamos en un vector destinado a ello.

Posteriormente, pasamos a calcular la media de las profundidades obtenidas de todos los tweets. Para ello, comprobamos que el vector de profundidad no esté vacío y llamamos a un método que recorre el vector de enteros con las profundidades de todos los tweets y las suma, guardándolas en

un acumulador y al final devuelve la suma total entre el tamaño total del vector, es decir, devuelve la media de todas ellas como valor de tipo double con dos decimales. Solo queda hallar la varianza y hacer la raíz para obtener la desviación típica de las profundidades, y así poder estudiar la dispersión de los valores con el valor medio, y comparar la profundidad de los tweets como parte de la calidad lingüística.

Indicador 3. Porcentaje de tweets que contienen oraciones subordinadas

Gracias al análisis sintáctico del tweet, podemos estudiar el contenido del mismo y valorar si la aparición de oraciones subordinadas, nos sirven para determinar si un usuario tiene mayor calidad lingüística que otro al considerarse un elemento lingüístico que aporta mayor riqueza a la frase, y creemos que abundan en lenguajes no convencionales.

La manera de detectar si un tweet contiene una oración subordinada es detectando la aparición de las etiquetas del parser SBAR (cláusula introducida por una conjunción subordinada) o SBARQ (pregunta directa introducida por la palabra o sintagma que empieza por wh). Para ello, dividimos el tweet en un árbol de etiquetas y comprobamos la aparición de estas dos. Puede haber más de una oración subordinada dentro de un tweet, pero solo nos interesa que haya una para poder decir que el tweet en conjunto contiene una oración subordinada.

A continuación muestro una frase subordinada adjetiva explicativa, que da una explicación adicional sobre el antecedente, que es la palabra a la que se refiere el pronombre relativo “who” y su salida en el parser.

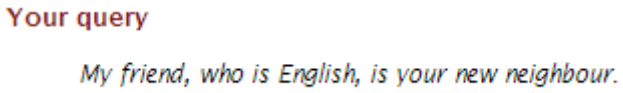


Figura 59: Frase a analizar con el parser

Parse

```
(ROOT
  (S
    (NP
      (NP (PRP$ My) (NN friend))
      (, ,)
      (SBAR
        (WHNP (WP who))
        (S
          (VP (VBZ is)
            (ADJP (JJ English)))))
      (, ,))
    (VP (VBZ is)
      (NP (PRP$ your) (JJ new) (NN neighbour)))
    (. .)))
```

Figura 60: Frase analizada utilizando el parser

¿Por qué elegimos estudiar los sintagmas adverbiales y adjetivales?

Es importante estudiar determinados sintagmas que aparecen en los tweets, y comprobar si son estadísticos suficientes para determinar la naturaleza de las frases, y ver si determinadas cuentas se decantan por el uso de unos sintagmas más que otros. En nuestro caso, hemos decidido calcular la cantidad de sintagmas adverbiales y adjetivales, dado que pensamos que resulta de mayor uso para un usuario que escribe con un registro más elaborado. Por eso no hemos hecho hincapié en el estudio de sustantivos y verbos, al abundar en un lenguaje más sencillo, aunque utilizado por todos y no habría forma de diferenciar diferentes tipos de cuentas.

Indicador 4. Número medio de sintagmas adverbiales

El estudio de este constituyente sintáctico resulta de interés al estar formado por un adverbio, el cual informa sobre las circunstancias en las cuales transcurre el verbo desempeñando la función de complemento circunstancial. En un lenguaje más sencillo y simple, deberían aparecer en menor medida al centrarse en el mensaje y no en un lenguaje complicado.

La forma de identificar el sintagma adverbial es analizar el tweet, dividirlo en un array de árboles y comprobar si la etiqueta sintáctica coincide con “ADVP”. Utilizamos un contador específico para contabilizar este sintagma para cada usuario, y así poder calcular el número medio de la forma: número total de sintagmas adjetivales / número total de tweets escritos en inglés y que no son RTs.

Indicador 5. Número medio de sintagmas adjetivales

Al igual que sucede con los sintagmas adverbiales, otro de los sintagmas que hemos elegido para comprobar la calidad de un usuario en Twitter es el sintagma adjetival. Son las agrupaciones de palabras en torno a un adjetivo, que funciona como núcleo o la palabra de mayor jerarquía de todas ellas, por lo que la aparición de estos sintagmas constituirá al núcleo con más relaciones sintácticas, y tendrá connotaciones positivas en cuanto a la calidad del tweet en cuestión. La riqueza de estos sintagmas pensamos que es abundante en cuentas de usuarios que escriben con especial atención a su lingüística, y con esa idea se calcula este identificador.

El parser proporciona "ADJP" como etiqueta sintáctica para los sintagmas adjetivales. Nuestro propósito es calcular el número medio, y lo hacemos de la siguiente forma:

Número total de sintagmas adjetivales / número total de tweets escritos en inglés y que no son RTs.

4.5.5.1.2. Registro linguístico

En esta parte se encuentran los identificadores basados en el registro lingüístico de los tweets, calculando el porcentaje de palabras distintas usadas en los tweets, y la distribución del uso de las últimas 20.000 palabras del Wiktionary, para el estudio de la variedad de palabras utilizadas en la cuenta de Twitter.

Dependiendo del tipo de usuario en Twitter, el modo de usar la lengua es diferente en un contexto que en otro. Hay usuarios que no hacen hincapié en las palabras utilizadas en Twitter, y otros que por prestigio o simplemente con más nivel cultural, les interesa expresarse con frases y palabras elaboradas. Es por ello que nos centramos en el registro lingüístico que tienen cada uno de los 6 tipos de cuentas, comparando las palabras del tweet con las palabras de la lista del Wiktionary en inglés, y comprobar el nivel de cada uno y ver si gracias a los dos indicadores de esta sección es posible diferenciarlas.

Como mencionamos anteriormente, utilizados el Wiktionary, ya que hacen un recuento de las palabras más relevantes en un idioma, y de esta forma podemos analizar el conjunto de palabras de los tweets posteados por cada usuario de Twitter. Para ello, guardamos cada una de las 36663 palabras en un fichero llamado Wiktionary.txt, donde el formato se encuentra explicado en el apartado 2.4.8. Se presenta parte del contenido del Wiktionary en la siguiente imagen:

1	the	56271872
2	of	33950064
3	and	29944184
4	to	25956096
5	in	17420636
6	i	11764797
7	that	11073318
8	was	10078245
9	his	8799755
10	he	8397205
11	it	8058110
12	with	7725512
13	is	7557477
14	for	7097981
15	as	7037543
16	had	6139336
17	you	6048903
18	not	5741803
19	be	5662527
20	her	5202501

Figura 61: Estructura del fichero Wiktionary.txt

Funcionamiento de cómo se comprueba si las palabras del tweet se encuentran dentro de las palabras del Wiktionary, con tres sencillo pasos:

(1) **(2)**

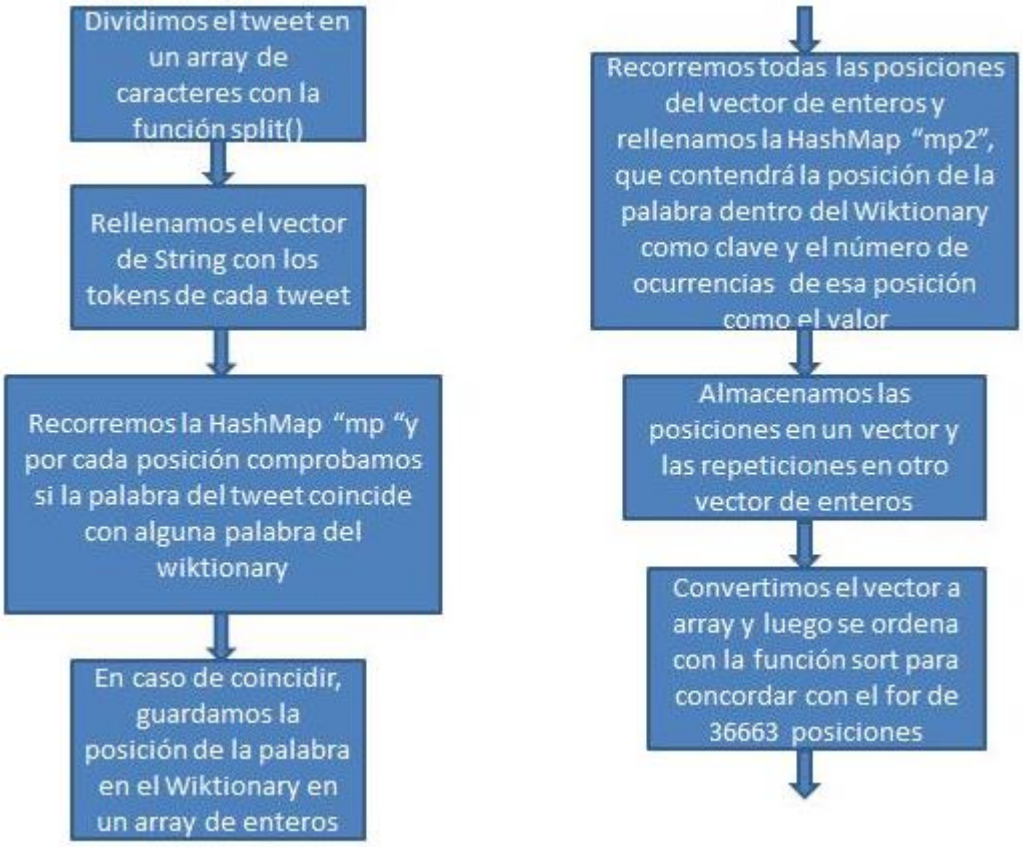


Figura 62: Pasos para comprobar si las palabras del tweet aparecen en el fichero Wiktionary.txt (Parte 1)

(3)

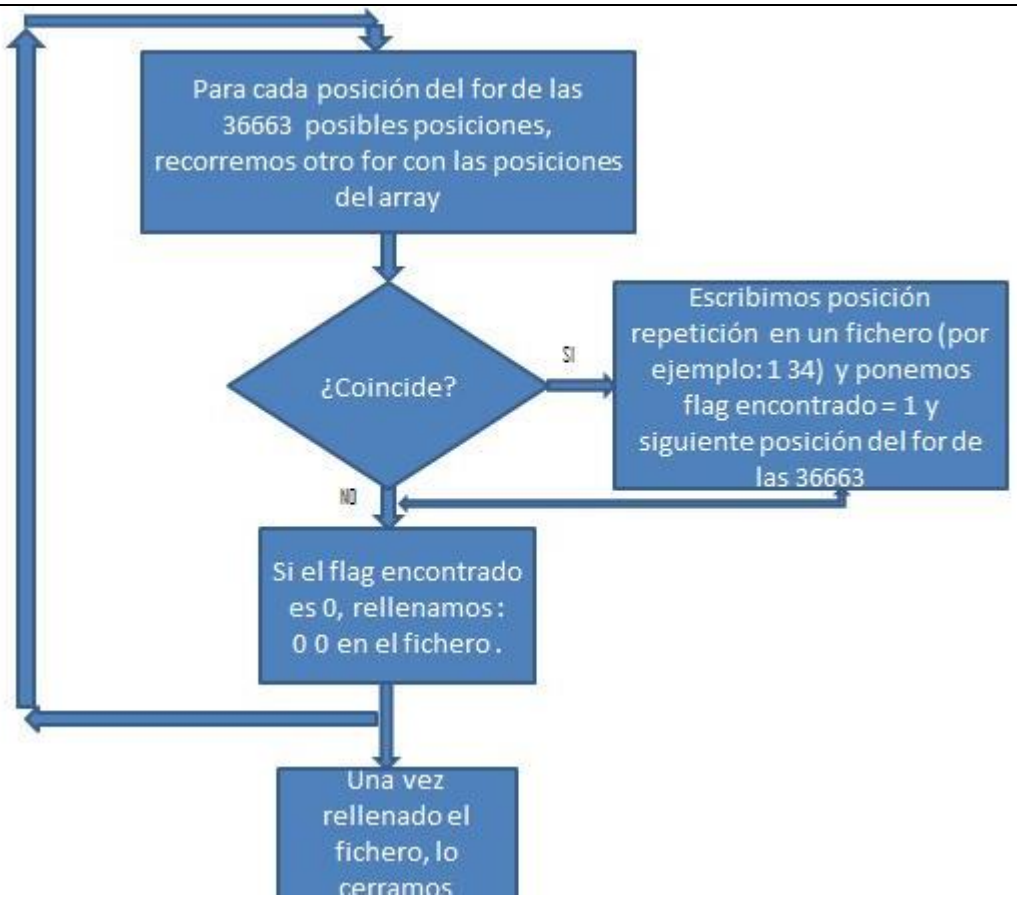


Figura 63: Pasos para comprobar si las palabras del tweet aparecen en el fichero Wiktionary.txt (Parte 2)

Con los pasos (1) y mitad del (2) podemos calcular el identificador 6. Para el identificador 7 es necesario completar los 3 pasos anteriores, ya que necesitamos el fichero con la información de las posiciones y el número de repeticiones de esas palabras.

Indicador 6. Porcentaje de las últimas 20.000 palabras distintas del Wiktionary que se usan en los tweets

En nuestro fichero llamado Wiktionary.txt se encuentran todas las palabras ordenadas de mayor frecuencia de uso a menor, por lo que vamos comparando si las palabras del tweet se encuentran en el fichero y hallamos el porcentaje. Se quiere comprobar que las cuentas bots utilizan un registro de palabras más común, y que, por tanto, se diferencia del resto de cuentas, como por ejemplo de las cuentas escritas por humanos, y en especial de las cuentas de calidad, las cuales comprobaremos si es verdad que usan un vocabulario más rebuscado, con una mayor variedad de palabras.

Al haber en torno a 40.000 palabras en la lista del Wiktionary, hemos escogido las últimas 20.000, alrededor de la mitad siendo un número razonable como parte del estudio del registro lingüístico, y suficientes para llevar a cabo la investigación. Debido a la ordenación de las palabras en nuestro fichero, nos hemos centrado en las últimas palabras al tener una frecuencia de uso menor, y por tanto, tratar de averiguar si los usuarios utilizan palabras más elaboradas. En el caso de haber seleccionado las primeras palabras, al ser de uso más común y coloquial, todos los usuarios obtendrían porcentajes altos, incluso las personas más intelectuales, ya que las palabras corrientes son usadas por todos, lo que hubiera dificultado el experimento cuando tuviéramos que diferenciar unas cuentas de otras.

Para el cálculo del porcentaje de palabras distintas del Wiktionary que aparecen en los tweets, primero hay que contabilizar el número de palabras distintas, y para ello, ya tenemos un vector de enteros con las posiciones de las palabras en el Wiktionary, ya comentado en el diagrama de flujo anterior. Ahora solo queda contabilizar las palabras usadas en las últimas 20.000 posiciones ya que son las que verdaderamente nos interesan, y lo hacemos de la siguiente manera:

```
for(for_posi_=0;for_posi_<posi.size();for_posi_++){
    if(posi.get(for_posi_)>=16663&&posi.get(for_posi_)<=36663){
            cont_20000_dist++;
    }
}
```

Porcentaje de las últimas 20.000 palabras distintas del Wiktionary utilizadas en los tweets = (cont_20000_dist * 100) / 20.000

Indicador 7. Desviación típica de la distribución del uso de las últimas 20.000 palabras del Wiktionary

Hacemos el análisis de datos agrupados en intervalos. Estos datos que analizamos son las posiciones de las palabras de los tweets en el Wiktionary, con sus respectivas repeticiones, a diferencia del anterior identificador en donde no se tenían en cuenta el número de veces en que se repiten las palabras cultas, sino solo cuántas coincidían con las últimas 20.000 del Wiktionary.

Procedemos a identificar cada intervalo con su marca de clase correspondiente, denotada como "xi", tal y como se aprecia en la siguiente tabla:

Intervalos	xi
[1,500]	250
(500,1000]	750
(1000,1500]	1250
(1500,2000]	1750
(2000,2500]	2250
(2500,3000]	2750
(3000,3500]	3250
(3500,4000]	3750
(4000,4500]	4250
(4500,5000]	4750
(5000,5500]	5250
(5500,6000]	5750
(6000,6500]	6250
(6500,7000]	6750
(7000,7500]	7250
(7500,8000]	7750
(8000,8500]	8250
(8500,9000]	8750
(9000,9500]	9250
(9500,10000]	9750

Intervalos	xi
(10000,10500]	10250
(10500,11000]	10750
(11000,11500]	11250
(11500,12000]	11750
(12000,12500]	12250
(12500,13000]	12750
(13000,13500]	13250
(13500,14000]	13750
(14000,14500]	14250
(14500,15000]	14750
(15000,15500]	15250
(15500,16000]	15750
(16000,16500]	16250
(16500,17000]	16750
(17000,17500]	17250
(17500,18000]	17750
(18000,18500]	18250
(18500,19000]	18750
(19000,19500]	19250
(19500,20000]	19750

Figura 64: Intervalos y marcas de clase para el indicador 7

Pretendemos estudiar la distribución del uso de las últimas 20.000 palabras de una lista de 36.663 palabras en inglés, almacenadas en un fichero de texto y sacadas de un proyecto de diccionario libre llamado Wiktionary. Se emplean 40 acumuladores, donde cada uno de ellos almacena las repeticiones de 500 palabras, por lo que 500 * 40 = 20.000 palabras en total que se analizan (las últimas 20.000 del Wiktionary). Dada la agrupación de las palabras del Wiktionary por sus repeticiones, se podrá ver como una distribución de probabilidad, donde:

- S = número total de repeticiones de las últimas 20.000 palabras del Wiktionary.
- F (500) = P(X<=500) = número de repeticiones de las palabras entre 1 y 500: (repe_1_500)/S.

El proceso consiste en crear 40 condiciones, donde cada una de ellas comprueba si el número está comprendido entre 500 valores posibles, para conocer cuáles de las últimas 20.000 palabras se ha utilizado y sus repeticiones correspondientes. Después leemos del fichero el primer token de cada línea, que es donde aparece la posición, y comprobamos en cuales de las 40 condiciones se encuentra, acumulando las repeticiones en el acumulador correspondiente. De modo que ya tenemos los acumuladores por agrupaciones de 500 palabras, y ahora procedemos a calcular el identificador correspondiente para este apartado.

Para calcular la desviación típica de la distribución del uso de las últimas 20.000 palabras, primero debemos calcular la media. Para ello debemos definir el punto medio de cada agrupación y multiplicarlo por el acumulador correspondiente, es decir, resolvemos la siguiente ecuación:

Media = acumulador_0_499 * 250 + acumulador_500_999 * 750+acumulador_1000_1499 * 1250 +…/S

Una vez obtenida la media, procedemos a calcular la varianza de la distribución:
Varianza = E[(x-E[x])²]== (250-media)² * acumulador_0_499 + (750-media)² * acumulador_500_999 * 750 + (1250-media)² * acumulador_1000_1499../S
Desviación típica = raíz (varianza). En lenguaje java es: Math.sqrt (varianza).

Representación gráfica de las últimas 20.000 palabras a través del histograma

Para escribir el documento de Microsoft (Excel en nuestro caso) se ha utilizado el Api de Java "Apache POI"[25]. Básicamente lo que hemos hecho ha sido crear un archivo con nombre "Histograma_nombre_usuario.xls", cuyo contenido interior son dos columnas: la primera está creada para guardar cada agrupación de posiciones de las palabras del fichero Wiktionary (de 500 en 500), y la segunda, el número de repeticiones de esa agrupación, que lo tenemos gracias a los 40 acumuladores calculados anteriormente. Para finalizar, realizando unos pasos con el programa Excel obtenemos el histograma para cada cuenta de usuario:

Seleccionamos la pestaña Insertar -> Columna -> Tipo de gráfico: columna en 2-D

A continuación se muestran dos ejemplos de cuentas de twitter: la primera es un medio de comunicación y al segunda un bot. Se pretende comparar ambos histogramas y observar la distribución de los datos, como parte de la detección de la calidad de un usuario en Twitter mediante la variedad de palabras usadas en los tweets.

- **Gráfico en Excel de las palabras repetidas del Wiktionary en intervalos de 500 palabras, de una cuenta de tipo periódico (time):**

Aquí mostramos un ejemplo del histograma generado con Excel[26] de una cuenta de Twitter de periódico, donde las primeras posiciones son las más usadas, pero a medida que aumentamos de posición de palabra en el Wiktionary, se ve más reducido el número de palabras usadas pero hay una cantidad considerable de ellas, por lo que se aprecia que se trata de una cuenta de Twitter que utiliza un registro más intelectual, si lo comparamos con el resto de cuentas como por ejemplo una cuenta bot, analizada posteriormente.

[25] http://poi.apache.org/
[26] Programa desarrollado y distribuido por Microsoft, utilizado normalmente en tareas financieras y contables usando hojas de cálculo.

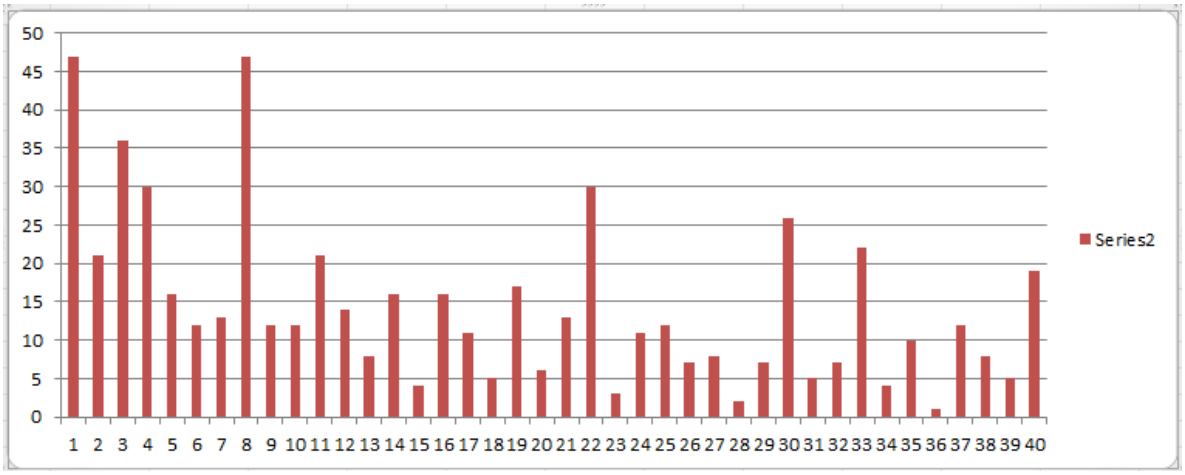


Figura 65: Histograma de las palabras repetidas del Wiktionary en intervalos de 500 (cuenta de periódico)

- **Gráfico en Excel de las palabras repetidas del Wiktionary en intervalos de 500 palabras de una cuenta de tipo bot (@hypem_charts):**

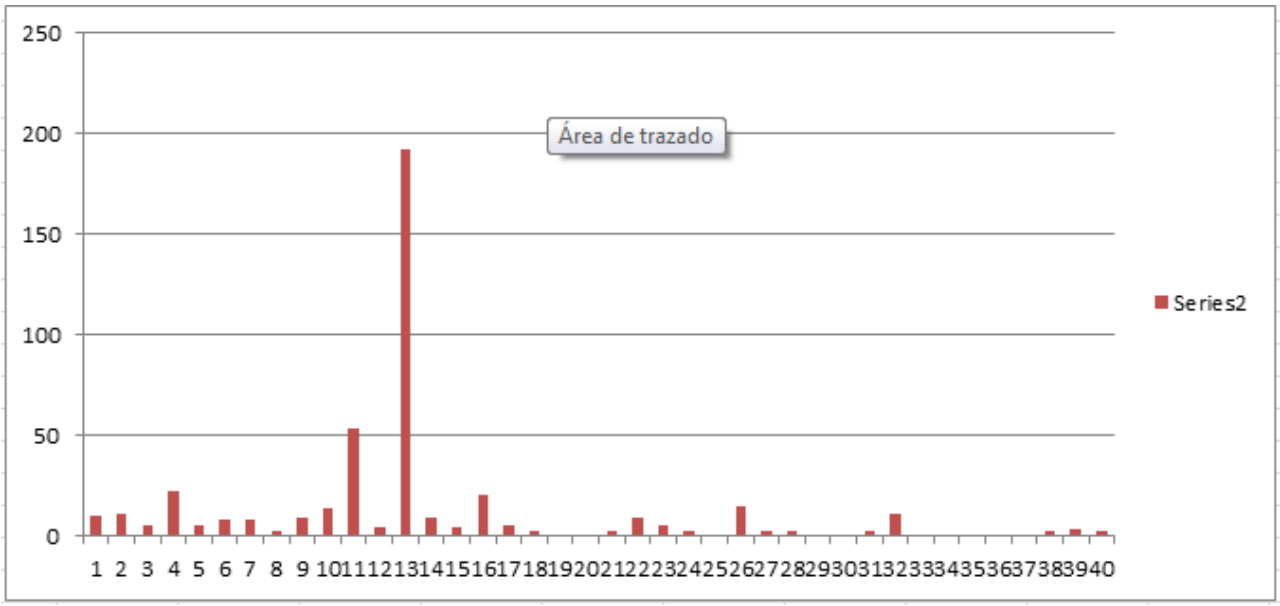


Figura 66: Histograma de las palabras repetidas del Wiktionary en intervalos de 500 (cuenta bot)

Una cuenta bot utiliza muy pocas palabras de las últimas 20.000 palabras del Wiktionary, por lo que mediante el histograma realizado con Excel se puede determinar si se trata de una cuenta de calidad lingüística o de una cuenta de robot con uso limitado de palabras instruidas.

4.5.5.2. Indicadores generales

Otro tipo de indicadores son los que hemos llamado generales, ya que contienen información característica de la cuenta de twitter, que hemos obtenido utilizando las APIs de Twitter. Tratamos de averiguar si gracias al número de RTs que tuvieron, la cantidad de hashtags y algunos más indicadores del mismo estilo, nos pueden servir para poder identificar tipos de cuentas en Twitter.

Indicador 8. Porcentaje de tweets que son RT

Con el número de RT pretendemos averiguar si nos puede dar información sobre la cuenta de Twitter, y cuál de los 6 tipos de cuentas analizadas retuitea más. Depende de la finalidad de cada usuario, tendrá unos determinados comportamientos u otros.

Las cuentas de calidad y de periódicos suelen hacer RT para dar a conocer determinados artículos de opinión, de investigación. Creemos que cuantos más RT hagan los profesores de universidad a personas que se encuentran dentro de su misma universidad como investigadores y demás personal, mayor captación de followers[27] tendrá aquellos a los que se hace RT, lo que conlleva mayor reputación social, y en cierta medida mejora tu credibilidad socialmente. Pensamos que el objetivo de las cuentas de Twitter de periódicos es captar lectores y hacer llegar sus titulares a todo el mundo, y por ello hacen RT a personas que les pueda interesar dentro de su entorno, mientras que las cuentas bots o fake, solo les interesa escribir a todo tipo de gente para parodiar (en el caso de los fake) o vender algún producto y enviar spam (en el caso de los bots), de modo que emiten pocos RT.

Indicador 9. Porcentaje de tweets que contienen menciones

Este identificador se centra en las menciones en el cuerpo del tweet. Nos parece un identificador interesante ya que al postear un tweet sobre tu estado personal o cualquier otra finalidad, al mencionar a otro usuario se pueden sacar determinadas conclusiones sobre la calidad del usuario, o eso pretendemos en este proyecto. Se quiere comprobar si los usuarios aleatorios, al ser personas escogidas al azar, twittean de acuerdo a una finalidad más común como estado personal, actividades que se han hecho o se dejan de hacer, mientras que otros usuarios como los de cuentas de calidad o medios de comunicación, se centran en el contenido del mensaje mandando información útil, y no tanto en mencionar a otras personas para comentar aspectos personales. También queremos comprobar cómo se comportan las cuentas automatizadas como son los bots, y si estas cuentas escogidas por nosotros sueles mencionar a determinados usuarios para enviarlos spam de todo tipo.

[27] Término inglés que significa seguidores, y son los usuarios que nos siguen en Twitter.

Indicador 10. Porcentaje de tweets que contienen hashtags

Los hashtags permiten diferenciar, destacar y agrupar una palabra o tópico específico en Twitter. También son útiles para obtener resultados de búsqueda. Por todo ello, hemos decidido conocer qué usuarios utilizan mayor número de hashtags en sus tweets, y comprobar si es un identificador válido para ayudarnos a saber la calidad de un usuario en Twitter, y poder diferenciar unos tipos de cuentas de otras.

Entendemos que las cuentas verdaderas deberían crear sus propios hashtags con la intención de convertirlos en temas del momento, debido a su gran popularidad que pueden alcanzar en apenas algunas horas, dado el gran número de seguidores que poseen. Las cuentas de calidad y de periódicos pensamos que utilizan hashtags para promover su contenido, agrupar sus seguidores y transmitir noticia e ideas. Sin embargo las cuentas bots utilizadas no están escogidas para utilizar hashtags para hacer trending topics, más bien usan enlaces para sus fines programados.

Indicador 11. Porcentaje de tweets que contienen enlaces

Con el análisis de este identificador pretendemos conocer aquellos usuarios que postean referencias a otros recursos, pueden ser imágenes subidas a Facebook o enlaces redirigidos a otras páginas webs. La forma de detectar un enlace se hace comprobando el comienzo de las letras de cada palabra del tweet, y si coincide con “http” lo contabilizamos como enlace.

Queremos comprobar si se cumple la hipótesis de introducir enlaces en los tweets los medios de comunicación, para facilitar al seguidor la información leída en los pocos caracteres que permite Twitter. Es interesante conocer si las cuentas bots twittean enlaces como forma de propagar spam.

Aquí un ejemplo de una cuenta de periódico: @nytimes
Se observa un titular corto, y el enlace al final del tweet.

Figura 67: Ejemplo de cuenta de periódico tuiteando un titular corto junto con un enlace

Indicador 12. Porcentaje de tweets que son conversación

Con este identificador tratamos de averiguar los usuarios que establecen conversaciones en Twitter. Cada usuario tiene su propia finalidad a la hora de Twittear, así por ejemplo un medio de comunicación se deberá de centrar en twittear su mensaje cada cierto tiempo, sin contestación a otra persona por regla general, mientras que cuentas de famosos, usuarios corrientes de la calle posiblemente darán más juego en Twitter en cuanto a establecer conversaciones con otros usuarios. Las cuentas bots elegidas no son del tipo conversacionales por lo que se espera un porcentaje bastante bajo.

Aquí un ejemplo de una cuenta de Twitter que pertenece al grupo de usuarios que hemos definido como de calidad, donde se aprecia el uso de la conversación como forma de comunicarse en Twitter:

Figura 68: Ejemplo de conversación en usuario de calidad

Indicador 13. Número total de usuarios que siguen en Twitter, más conocido como followees[28]

Nos interesa conocer el número de followees de cada usuario para saber el número al que está suscrito a sus tweets como un seguidor, y poder compararlo con el número de seguidores que tiene y ver si se pueden diferenciar unos tipos de cuentas con otras.

Normalmente las personas muy conocidas suelen seguir a poca gente si lo comparamos con el número de seguidores que tienen, que en muchos casos suele ser de millones. En este caso creemos que deberían pertenecer a nuestras cuentas verdaderas que son todas personalidades conocidas, y a las cuentas de medios de comunicación que también son bastantes conocidas. Aunque también puede darse el caso de que mantengan buenas relaciones entre periódicos de todo tipo, y entre profesores de universidad para las cuentas de calidad, siguiéndose mutuamente.

Por el contrario, si el usuario sigue a muchos usuarios y tiene pocos seguidores, debería tratarse de cuentas bots programadas con la intención de captar seguidores, donde si en un tiempo no le seguir, dejan de seguir a estas cuentas.

Indicador 14. Número total de followers en Twitter

[28] Término inglés que significa personas a las que seguimos, utilizado en Twitter para conocer la gente a la que seguimos por el interés que tenemos en conocer lo que twittea.

El número de followers puede denotar la influencia que tengas en la red social Twitter. Se espera que las cuentas de periódicos tengan un número elevado de seguidores, al igual que las verdaderas, dada su popularidad y la necesidad de estar informados de las noticas actuales en tiempo real, y conocer las actividades en la vida real de las personas conocidas del país y del mudo. Las cuentas de calidad deben tener seguidores propios de la universidad o de su círculo de investigación, aunque siempre tiene que haber excepciones. Las cuentas que tienen que tener un número de seguidores más reducido deberían ser las aleatorias escogidas, al ser menos conocidas en esta red social. Las cuentas bots deberían de presentar pocos seguidores ya que poca gente querrá seguir a estas cuentas, y encima si envían enlaces maliciosos mucho menos. Pero también se puede dar el caso de cuentas que traten de seguir a todo el mundo, y consigan su propósito de captar followers, y así poder propagar sus contenidos de spam, e incluso por mensajes directos.

4.5.5.3. Indicadores temporales

Este apartado es referente al estudio de la frecuencia de twitteo, de modo que analizamos el tiempo transcurrido en segundos entre un tweet y su anterior. Este cálculo nos sirve para calcular dos indicadores temporales: el intervalo mayor de entre todas las diferencias en segundos por cada dos tweets, y la desviación típica de esas diferencias para ver la dispersión de los datos.

Indicador 15. Cálculo del intervalo máximo entre un tweet y su anterior

En primer lugar, se ha obtenido la fecha en que se posteó cada tweet que vamos a analizar. Por cada dos tweets consecutivos, llamamos a la función dif_segundos(String s1, String s2), que recibe como parámetro las fechas de cada tweet en formato String. Para calcular la diferencia de segundos entre las dos fechas, hacemos uso de SimpleDateFormat que nos permite construir nuestro propio formato, donde las fechas son construidas tomando como referencia un String que especificará un patrón para parsear las fechas [31]. El formato elegido es el siguiente: yyyy-MM-dd HH:mm:ss, es decir, año-mes-día hora:minutos:segundos, un ejemplo de fecha es la siguiente: 2014-1-14 20:01:40. Lo parseamos para convertirlo a tipo Date. Necesitamos la instancia del calendario gregoriano para cada fecha, y posteriormente, establecer la fecha del calendario con el date obtenido anteriormente. Con getTimeInMillis() obtenemos los milisegundos desde el 1 de enero de 1970 para cada fecha, y finalmente calculamos la diferencia de milisegundos entre las dos.

```
public static long dif_segundos(String s1, String s2){
    java.util.Date date1 = null;
    java.util.Date date2 = null;

    long milisegundos1, milisegundos2, diferencia;

    SimpleDateFormat s = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

    try {
        date1 = s.parse(s1);
    } catch (ParseException e) {
        e.printStackTrace();
    }
    try {
        date2 = s.parse(s2);
    } catch (ParseException e) {
        e.printStackTrace();
    }

    Calendar c1 = Calendar.getInstance();

    Calendar c2 = Calendar.getInstance();

    c1.setTime(date1);
    c2.setTime(date2);

    milisegundos1 = c1.getTimeInMillis();

    milisegundos2 = c2.getTimeInMillis();

    diferencia = milisegundos2 - milisegundos1;

    long dif_segundos =  Math.abs (diferencia / 1000);

    return dif_segundos;

}
```

Figura 69: Cálculo de la diferencia de segundos entre dos tweets

Una vez recorridas todas las fechas de los tweets y calculado la diferencia de segundos entre cada dos tweets y almacenando cada diferencia en un vector de tipo Long, lo que falta por hallar es la máxima diferencia de segundos de entre todos los valores calculados. Para ello, convertimos el Vector a un array con el método toArray(), y luego ordenamos ese array de forma descendente con Arrays.sort(array), por lo que el máximo valor se encontrará en la cima, es decir, en la posción 0 del array, y ya tenemos el intervalo máximo entre dos tweets consecutivos, lo que denominamos el gap máximo. Tomamos el logaritmo natural (de base e, conocido normalmente como logaritmo neperiano) para obtener representaciones gráficas más claras y poder distintar unos box-plots de otros.

Diagrama de flujo del cálculo de la diferencia de segundos entre dos tweets:

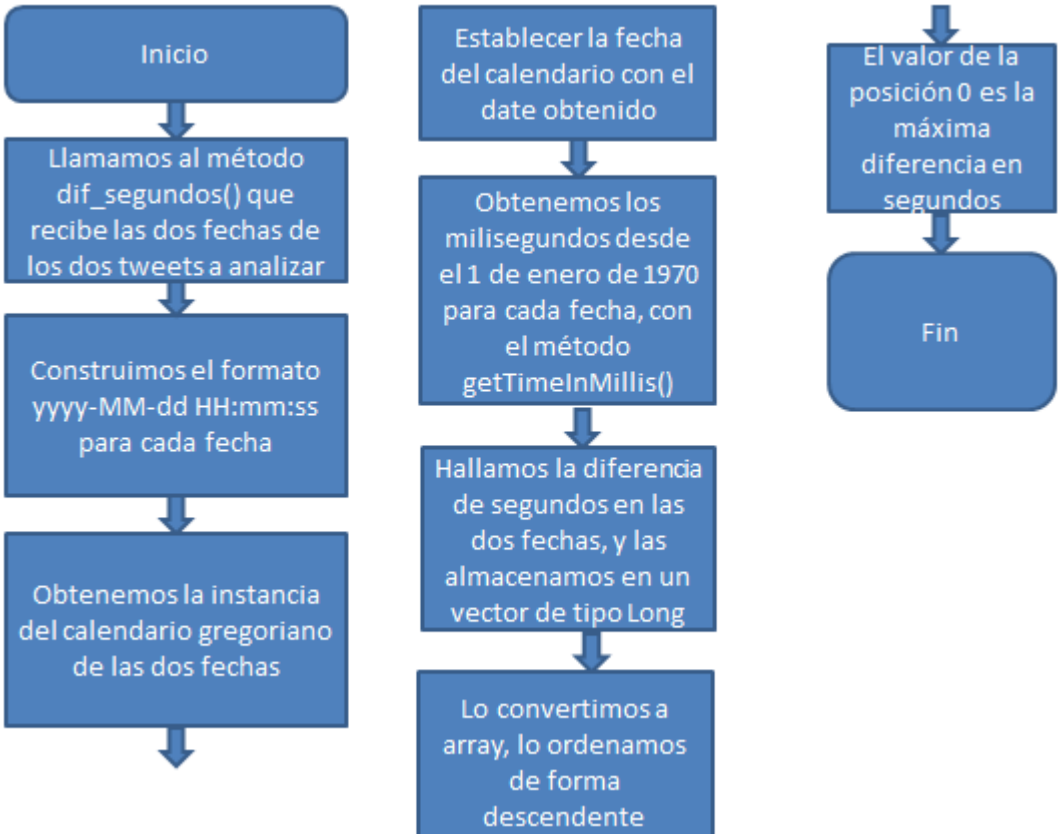


Figura 70: Diagrama de flujo del cálculo de la diferencia de segundos entre dos tweets

Indicador 16. Desviación típica de la diferencia de segundos entre un tweet y el siguiente

El diagrama de flujo es similar al anterior, pero en el tercero paso cambia: en este caso no nos interesa el intervalo máximo, sino calcular primeramente la media, por lo que por cada iteración del bucle en donde calculamos las diferencias en segundos entre cada dos fechas, acumulamos dichos valores y sacamos la media de la diferencia en segundos que hemos obtenido de entre todos los tweets, dividiendo dicho acumulador entre el tamaño total menos 1del vector fechas. Una vez obtenida la media, y acto seguido la varianza, solo queda calcular la desviación típica (en segundos) de entre cada dos tweets para conocer la dispersión de twitteo de cada cuenta de usuario. Tomamos el logaritmo natural (de base e, conocido normalmente como logaritmo neperiano) para obtener representaciones gráficas más claras y poder diferenciar unos box-plots de otros.

4.5.6. Módulo 6: Representación gráfica de los datos

En este apartado vamos a explicar cómo hemos representado los resultados obtenidos al analizar cada uno de los identificadores para todas las cuentas.

En primer lugar, creamos un script R necesario para generar los box-plots para cada identificador.

Lo tenemos que ejecutar en un sistema Linux en el que R estuviese previamente instalador (es el paquete estándar: r-base).

Una vez dados los permisos de ejecución para ejecutarlo desde el terminal, tendremos que hacer:

./script.r <fichero de datos> <fichero de salida [EPS]>

- El fichero de datos que hemos utilizado son los resultados de los identificadores para cada uno de los usuarios, es decir, para el identificador 1 tendremos el fichero "fichero_ident1.txt", que contiene 6 columnas ya que hay hemos elegido 6 tipos de cuentas de usuario. En cada columna aparecen los valores obtenidos de cada identificador para cada usuario.
- El fichero de salida con extensión eps[29] es el gráfico que generamos al ejecutar el script. Dicho archivo lo podemos abrir con cualquier visor de postscript como Ghostview[30].

Este escrip.t lo que hace es leer los datos del fichero en el lugar del argumento 1, salvamos una figura en postscript con el comando: postscript("nombrefichero.eps") , hacemos uso del comando boxplot donde utilizamos los datos leídos anteriormente del argumento 1 y le decimos que lea cada una de las columnas y genera un box-plot por cada una de ellas, Finalmente acabamos con dev.off().

Mediante los box-plots podemos ilustrar los datos obtenidos para cada identificador, estudiar su simetría, la distribución de los datos de cada columna y poder comparar unos box-plots con otros.

4.6. Implementación

[29] Formato de archivo de gráfico.
[30] Programa intérprete de documentos en formato PS (y también en PDF).

Se detalla la implementación de las diferentes clases que componen el proyecto y una explicación profunda de los atributos y métodos que componen cada clase.

4.6.1. Diagrama de clases

A continuación se muestra el diagrama UML de las clases que componen el proyecto y las relaciones entre ellas:

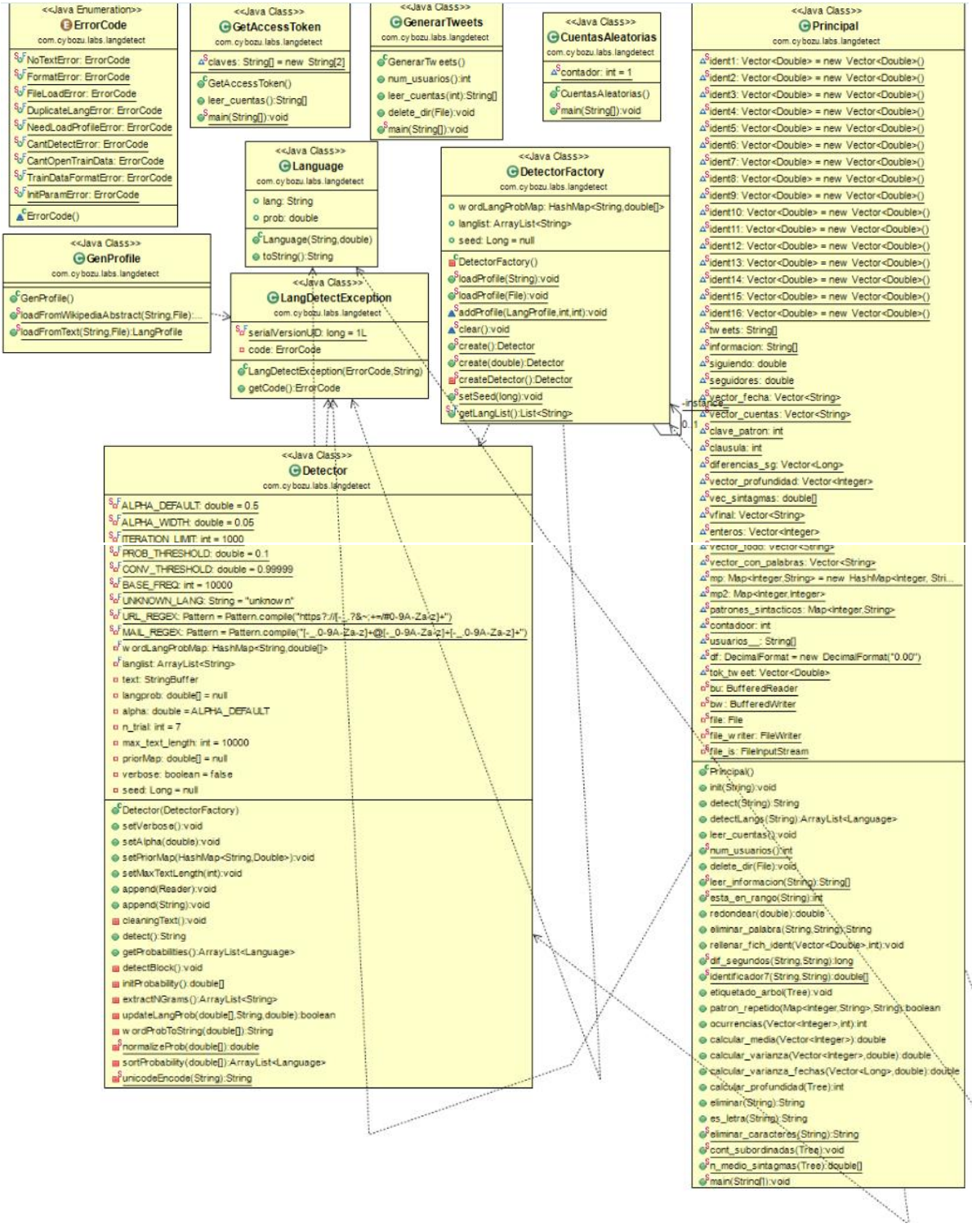


Figura 71: Diagrama de clases

4.6.2. Explicación detallada de las clases, con sus atributos y métodos

A continuación se va a especificar las clases que hemos implementado nosotros, con sus atributos y métodos. Se proporciona el código completo y bien documentado en los CDs[31] que proporcionaré para que los tenga la Universidad, además las clases utilizadas por los paquetes de algunos módulos ya vienen con documentación, pero aun así, lo comentamos brevemente cada uno de ellos:

Clase	Descripción/Atributos/Métodos
Principal	Clase principal del proyecto encargada de generar los 16 ficheros con los resultados de cada usuario de Twitter, para posteriormente poder generar los 16 box-plots. También genera un fichero para cada cuenta con la información detallada de los 16 identificadores y un fichero log por si hubiera generado algún tipo de error durante la ejecución del código. **Atributos:** • **bu:** objeto de la clase BufferedReader encargado de poder leer líneas completas del fichero. • **bw:** objeto de la clase BufferedWriter encargado de poder escribir líneas completas del fichero. • **clausula:** de tipo entero, sirve para conocer si hay una oración subordinada en el tweet. • **clave_patron:** valor entero para insertar la posición del patrón sintáctico en la HashMap. • **df:** define el formato de números hasta 2 decimales. • **diferencias_sg:** vector de tipo Long para almacenar las diferencias de segundos entre cada dos tweets consecutivos. • **enteros:** vector de enteros donde se encuentran las repeticiones de las palabras del tweet. • **file:** objeto de la clase FileReader para abrir el fichero de texto para leer. • **file_is:** objeto de la clase FileInputStream encargado de leer byte a byte el contenido del fichero.

[31] Disco óptico utilizado para almacenar datos en formato digital, consistentes en cualquier tipo de información (audio, imágenes, vídeo, documentos y otros datos).

	• **file_writer:** para conocer cuál es el fichero en el que queremos escribir. • **ident1-ident16:** vectores de tipo double encargados de almacenar los 240 valores correspondientes a los resultados de cada cuenta para cada identificador correspondiente. • **información:** vector de String rellenado con las líneas leídas del fichero usuario_informacion.txt. • **mp:** HashMap que contiene la posición y la palabra de la lista de palabras del Wiktionary. • **mp2:** HashMap que contiene la posición como clave y el número de repeticiones de esa posición como valor, ambas de tipo enteras. • **patrones_sintacticos:** HashMap que consta de una clave de tipo Integer y un valor de tipo String, la cual recorremos y vemos si uno de los valores coincide con el "patron" pasado como parámetro para comparar. • **seguidores:** contabilizamos el número de seguidores que tiene cada usuario. • **siguiendo:** contabilizamos el número de cuentas a las que sigue cada usuario. • **tweets:** array de String encargado de almacenar todos los tweets en cuestión. • **usuarios__:** array de String que contiene el nombre de los 240 usuarios de twitter. • **vec_sintagmas:** array de sintagmas de 7 posiciones, cada una de las cuales pertenece a un sintagma diferente. • **vector_con_palabras:** vector de tipo String que contiene las palabras que forman el tweet. • **vector_cuentas:** vector que contiene las cuentas de Twitter leídas del fichero todas_cuentas.txt. • **vector_fecha:** vector de tipo String que contiene las fechas de todos los tweets de cada usuario. • **vector_profundidad:** vector de tipo entero que contiene las profundidades de los árboles sintácticos de cada tweet. • **vector_todo:** el cual va a contener las etiquetas sintácticas de cada tweet y las palabras del tweet. • **vfinal:** vector de String que contiene todas las palabras de los tweets para cada usuario, y nos sirve para calcular el porcentaje de palabras que se encuentran en el Wiktionary. **Métodos:** ○ **cont_subordinadas (Tree parse):** recibimos el árbol sintáctico por

	parámetro y creamos un array de árboles sintácticos (Tree en lenguaje java) y vamos comprobando si se encuentran las etiquetas SBAR o SBARQ en los valores de los árboles. En ese caso, incrementamos clausula. AL final con que clausula valga 1, ya sabemos que por lo menos ha habido un SBAR o SBARQ y, por tanto, la oración es subordinada. La función se llama recursivamente por cada Tree. o **dif_segundos (String s1, String s2):** calculamos la diferencia de segundos entre la fecha del primer parámetro y la fecha del segundo parámetro recibido. o **eliminar_caracteres (String cadena):** eliminamos los caracteres que el parser de la Universidad de Stanford no permite parsear. o **Identificador7 (String nombre_fichero, String nombre_usuario_parametro):** Método para calcular la media y varianza de la distribución del uso de las últimas 20000 palabras del Wiktionary. *** @param** nombre_fichero que vamos a leer las repeticiones * **@param** nombre_usuario_parametro para crear el histograma * **@return** array de dobule de 2 valores: posicion 0: media, posicion 1: varianza * **@throws** IOException o **leer_informacion(String usuario):** leemos del fichero con nombre pasado por parámetro y rellenamos el vector de String información, que nos interesa porque hay información relativa a cada tweet. * **@param** nombre de usuario del que vamos a leer de su fichero usuario_informacion.txt * **@return** Vector de String informacion rellenado con las líneas leidas del fichero usuario_informacion.txt o **main:** hace llamadas a todas las rutinas que componen nuestro programa. o **n_medio_sintagmas(Tree parse):** Dado el árbol sintáctico de tipo Tree, lo dividimos en un array de árboles y por cada árbol miramos si el valor coincide con las etiquetas de los sintagmas que puede tener el parser, e incrementamos la posición del array de sintagmas en función de la etiqueta leída. Se llama recursivamente para cada árbol hijo. Este es el convenio utilizado: * vec_sintagmas[0] --> Sintagma adjetival * vec_sintagmas[1] --> Sintagma adverbial * vec_sintagmas[2] --> Sintagma preposicional * vec_sintagmas[3] --> Sintagma nominal

	* vec_sintagmas[4] --> Sintagma verbal * vec_sintagmas[5] --> Sintagma conjuncional * vec_sintagmas[6] --> Sintagma cuantificador * **@param** parse que vamos a analizar * **@return** array de double con 7 posiciones o **num_usuarios ():** leemos los nombres de las cuentas de Twitter del fichero todas_cuentas.txt y las contabilizamos que es lo que nos interesa. * **@return** Número total de cuentas que hay en el fichero todas_cuentas. o **calcular_media(Vector <Integer> vector):** este método recibe como parámetro un vector con las profundidades de los tweets y realiza la media recorriendo cada posición y acumulando cada profundidad en una variable de tipo double. * @param Vector de enteros que va a contener las profundidades de todos los tweets en inglés y que no son RT. * @return Media de las profundidades calculada como la suma total de las profundidades entre el tamañoo total del vector. o **calcular_profundidad(Tree parse):** este método devuelve la profundidad del árbol sintáctico. * @param Árrbol sintáctico de tipo Tree a través del cual vamos a obtener su profundidad (parse) * @return Profundidad del Árbol sintáctico o **calcular_varianza(Vector <Integer> vector, double media):** este método recibe como parámetro un vector con las profundidades de los tweets y la media de dichas profundidades obtenidas con el método calcular_media. Recorremos el vector y a cada valor le resto la media y elevo el resultado al cuadrado (la diferencia al cuadrado) con la función Math.pow, donde el primer parámetro es la base (x-media) y el segundo el exponente (en este caso 2). El resultado de cada diferencia al cuadrado las acumulo y devolvemos ese acumulador entre el tamaño total del vector, lo que viene siendo la varianza. * @param Vector con las profundidades de los tweets y la media de dichas profundidades obtenida con el método calcular_media. * @param Media de las profundidades de los árboles sintácticos * @return Varianza de la profundidad del árbol sintáctico.

	○ **calcular_varianza_fechas(Vector <Long> vector, double media):** calcula la varianza de la diferencia de tiempo (en segundos) de entre cada dos tweets, ya que obtengo las fecha en que se twitteó cada tweet. La idea es ver la dispersión de twitteo de cada cuenta. * @param Vector con las diferencias de segundos entre cada dos tweets * @param Media de las diferencias de segundos que hemos obtenido de entre todos los tweets * @return Varianza de las diferencias de segundos entre cada dos tweets ○ **delete_dir(File dir):** obtenemos un array de File de todos los ficheros que contiene el directorio, y al recorrerlo si es directorio llamamos al método recursivamente para buscar sus ficheros, en caso de no ser directorio lo borro. * @param Directorio que vamos a borrar ○ **detect(String text):** método encargado de detectar el idioma de un texto * @param Texto para detectar el idioma * @return perfil de idioma del texto detectado, por ejemlpo devolvería "en" si fuera escrito el texto en inglés * @throws LangDetectException ○ **detectLangs(String text):** método para calcular la probabilidad de detección del idioma. Si es seguro, dará una probabilidad 0.99 * @param Texto para detectar el idioma y posteriormente calcular la probabilidad de detección de idioma * @return Probabilidad de detección de idioma, por ejemplo: [en:0.9999962916000225] * @throws LangDetectException ○ **eliminar(String cadena):** recibimos un String y vamos a procesarla de tal forma que construimos una nueva cadena de solo letras y dígitos * @param Cadena que vamos a analizar * @return Cadena que solo contiene letras y dígitos ○ **eliminar_palabra(String frase, String palabra):** recibimos el tweet y un enlace, y el tweet lo dividimos en tokens y creamos un tweet nuevo que será la concatenación de todas las palabras del tweet inicial a excepción del enlace recibido. * @param Tweet que vamos a dividir en tokens * @param Enlace que queremos borrar ○ **es_letra (String cadena):** eliminamos todo lo que no sea letra en la cadena pasada como argumento, para que el detector de idioma no de errores. Para

	ello la convertimos en un array de char y vamos comparando cada carácter si es letra o no. Solo concatenamos los caracteres que sí lo sean, construyendo una nueva cadena la cual devolvemos. * @param Cadena de String * @return Cadena que contiene únicamente letras o **etiquetado_arbol (Tree parse):** este método descompone el árbol sintáctico pasado por parámetro en un array de árboles llamando al método children(), y por cada uno de ellos obtenemos el etiquetado y lo convertimos a tipo string para así meterlo en el vector de string vector_todo, el cual va a contener las etiquetas sintácticas de cada tweet y las palabras del tweet. Se llama recursivamente al método hasta el final del array de árboles. * @param árbol sintáctico que vamos a procesar(parse) o **init(String profileDirectory):** cargamos los perfiles del directorio profiles * @param Directorio profiles * @throws LangDetectException o **leer_cuentas():** leemos los nombres de las cuentas de Twitter del fichero todas_cuentas.txt y los introducimosen el vector_cuentas * @throws IOException o **ocurrencias (Vector <Integer> i, int numero):** dados el vector de enteros y un número, se recorre el vector y vamos contabilizando el número de veces que aparece el número en ese vector. * @param Vector de enteros donde se encuentran las repeticiones de las palabras del tweet * @param número que vamos a analizar cuantas repeticiones hay dentro del vector * @return Número de ocurrencias del número pasado como parámetro en el vector de enteros i o **patron_repetido(Map<Integer, String> patrones_sintacticos, String patron):** Este método comprueba si el patrón sintáctico se encuentra en la HashMap como valor de tipo String. Dicha HashMap va a contener patrones sintácticos diferentes, nunca pueden estar repetidos. * @param HashMap que consta de una clave de tipo Integer y un valor de tipo String, la cual recorremos y vemos si uno de los valores coincide con el "patron" pasado como parámetro para comparar. * @param Patrón sintáctico que vamos a comprobar si se encuentra en la HashMap * @return true: si el patrón pasado como parámetro ya se encuentra en la

<table>
<tr><td></td><td>HashMap como valor, false: una vez recorrida la HashMap el patrón que queremos comprobar no se encuentra en dicha HashMap.
o redondear(double numero): redondeamos el número pasado como parámetro y devolvemos el número redondeado con dos cifras decimales
* @param numero a redondear
* @return número redondeado con dos cifras decimales

o rellenar_fich_ident(Vector<Double> v, int i): este método se llama tanta veces como identificadores haya, y por cada llamada recibe un vector con los valores del identificador correspondiente (240 valores) y rellena el fichero con 6 columnas:
6 tipo de cuentas:
* x1-Calidad
* x2-Bot
* x3-Fake
* x4-cuentas verdaderas
* x5-cuenta de profesores de universidad, investigadores
* x6-cuentas aleatorias sacadas con el Streaming API
Las posiciones serían estas: pero no escribo las posiciones sino los valores en esa posición
<table><tr><td>X1</td><td>X2</td><td>X3</td><td>X4</td><td>X5</td><td>X6</td></tr><tr><td>0</td><td>40</td><td>80</td><td>120</td><td>160</td><td>200</td></tr><tr><td>1</td><td>41</td><td>81</td><td>121</td><td>161</td><td>201</td></tr><tr><td>2</td><td>42</td><td>82</td><td>122</td><td>162</td><td>202</td></tr></table>
* @param Vector de tamaño igual al número de cuentas que vamos a analizar (240)
* @param Número de identificador a analizar
* @throws IOException</td></tr>
<tr><td>GetAccessToken</td><td>Clase encargada de generar la clave del usuario, necesario para la autentificación por Oauth.
URL autorizada:
Atributos
o claves: array de tipo String donde almacenamos las claves de la aplicación
Métodos
o leer_cuentas (): leemos del fichero AppTello.key que contiene las claves de la aplicación para sacar tweets y las almaceno en un array de claves de tipo</td></tr>
</table>

	String y lo retorno. * @return Array de tipo String con las claves de la aplicación * @throws IOException o **main**: Se consigue el token_request, abrimos un navegador para aceptar la aplicación y te da un número, copiamos y pegamos este número en el comando y se generará un fichero con la clave, introducimos el PIN que aparece en la URL y de esta forma hemos conseguido el token de acceso(accessToken.getToken()) y el token de acceso secreto(accessToken.getTokenSecret())
CuentasAleatorias	Clase que obtiene cuentas aleatorias utilizando el Streaming API, con el criterio de que twitteen al menos 10 tweets en inglés. Tiene varios métodos elaborados para la API de streaming. Todo lo que necesitas es tener una clase que implementa StatusListener. Twitter4J va a hacer la creación de un hilo, el consumo de la corriente. **Atributos:** **contador:** contador que nos sirve para parar la búsqueda al llegar a los 40 usuarios que necesitamos. **Métodos:** o **main:** usando onStatus(Status status) podemos conocer el autor del tweet cogido, y el idioma en que está escrito, y una vez obtenido cerramos el hilo interno compartid por todas las instancias de TwitterStream.
GenerarTweets	Se encarga de generar dos ficheros: uno con 1400 tweets para cada usuario leído del fichero "todas_cuentas.txt" y otro con la información general en Twitter como la fecha en que se posteó, si es RT, hashtag, enlace, conversación y su idioma. **Métodos:** o **delete_dir(File dir):** obtenemos un array de File de todos los ficheros que contiene el directorio, y al recorrerlo si es directorio llamamos al método recursivamente para buscar sus ficheros, en caso de no ser directorio lo borro. * @param Directorio que vamos a borrar 4.2. **leer_cuentas(int n):** Leemos del fichero las cuentas y las almacenamos en el array de String que luego devolvemos *@param Número de cuentas leídas anteriormente del fichero todas_cuentas * @return Array con todas las cuentas leídas del fichero

	* @throws IOException 4.3. **int num_usuarios():** leemos del fichero todas_cuentas y contabilizamos el número de líneas que hay, es decir, el número de cuentas de Twitter * @return Número de cuentas que están escritas en el fichero, una por cada línea
Detector	Clase encargada de detectar el idioma del texto solicitado. Toda la documentación aparece en el paquete "langdetect-09-13-2011" que se proporciona en los CDs.
DetectorFactory	Maneja la inicialización y los constructores de Detector (detallado en "langdetect-09-13-2011").
LangDetectException	Se encarga de la captura de excepciones que puedan surgir a lo largo del proyecto.
Language	Su finalidad es almacenar el idioma detectado y obtener la probabilidad de detección del idioma.
GenProfile	Carga el archivo de base de datos abstracta y generar su perfil lingüístico.

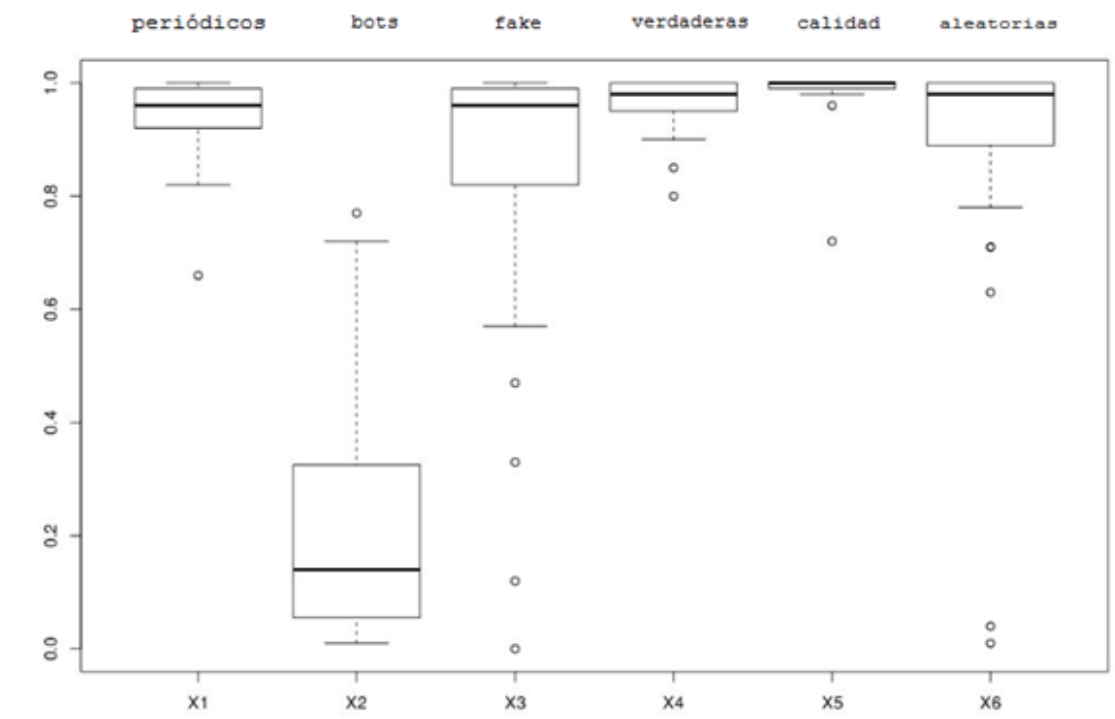
Tabla 4: Descripción, atributos y método de todas las clases

5. Análisis de los experimentos realizados

En este apartado nos centramos en analizar los resultados obtenidos para cada uno de los indicadores, que hemos calculado durante el desarrollo del proyecto. Para ello, hacemos uso de los box-plots, usando el script explicado en el apartado 2.4.9 para obtener las representaciones gráficas de los datos. Los agrupamos en tres categorías características: análisis lingüístico, análisis general de la cuenta de Twitter y análisis de la frecuencia de twitteo.

5.1. Análisis lingüístico

Indicador 1: Número de patrones sintácticos distintos en tweets dividido por el número de tweets



Y: Número de patrones sintácticos distintos en tweets / número de tweets
X: Tipos de cuentas

Figura 72: Box-plots del indicador 1

En el box-plot anterior se puede apreciar que el rango de patrones distintos de las cuentas bots oscilan entre 0,08 y 0,33, siendo la mediana de 0,15 aproximadamente, que refleja que hay 150 patrones distintos de los 1000 patrones posibles (sin contar RTs), o lo que es lo mismo, entre los 850 tweets que siguen un patrón, pueden seguir todos el mismo patrón y esa cuenta bot ha twitteado sin parar el mismo formato, o hay diferentes patrones a lo largo del tiempo, y por cada patrón twittea una serie de tweets y luego cambia, que es lo que hacen habitualmente. El hecho de que el box-plot para el tipo de cuenta bot no solape con ninguno de los demás box-plot, y tenga valores tan bajos, denota que es un indicador válido para diferenciar cuentas bots del resto.

Las cuentas fake son las que tienen algunos patrones sintácticos repetidos, quizás por utilizar tweets similares para parodiar a determinados famosos. Las aleatorias también repiten, porque diariamente se suele twittear muy parecido y sobre 1000 tweets se puede repetir algunos, pero en cualquier caso, insignificante en comparación con bots.

En realidad hay que fijarse en las cajas y ver que no se solapen, y nos permiten ver la dispersión de los valores con la mediana. A continuación, se muestra una tabla con los resultados de la mediana, donde claramente las cuentas bots tienen un valor más pequeño que el resto de cuentas:

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana	0,95	0,15	0,95	0,98	1,00	0,98

Tabla 5: Medianas del indicador 1

Indicador 2: Desviación típica de la profundidad del árbol sintáctico de cada tweet

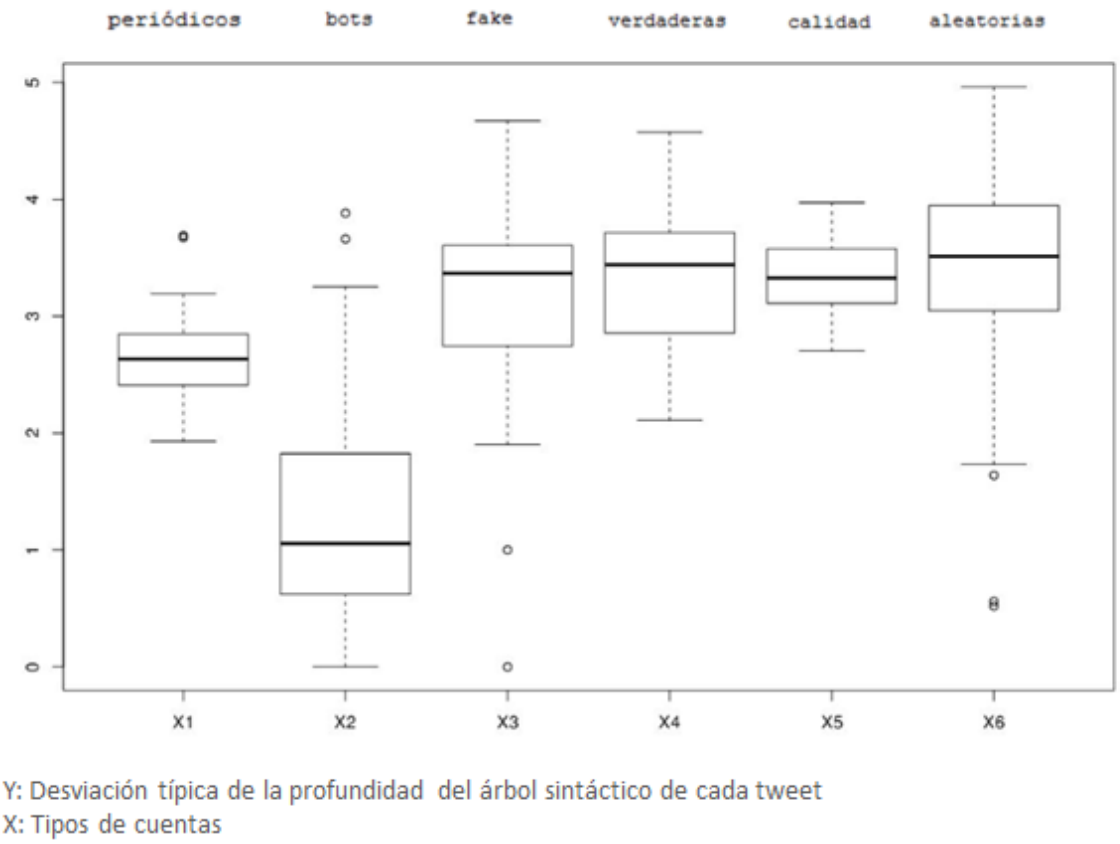


Figura 73: Box-plots del indicador 2

Mediante el gráfico que representa el diagrama de caja con los valores de las desviaciones típicas de la profundidad de los tweets para cada cuenta, contemplamos que el box-plot relacionado con las

cuentas bots se encuentra en valores más bajos con respecto a todas las demás. Y a su vez las cuentas de periódicos obtienen unos valores bastante bajos con respecto a las demás, excepto las bots. Gran causa la tiene que los periódicos tuitean titulares cortos de sus periódicos, acompañado de un enlace, por lo que se demuestra que no se puede extraer una profundidad significativa de sus tweets. Aunque el resultado característico con este gráfico es que si utilizamos valores como la mediana (Q2) del box-plot, claramente el valor de 1,2 de las cuentas bots está muy por debajo del resto, y puede ser un indicador útil para distinguir al resto de cuentas.

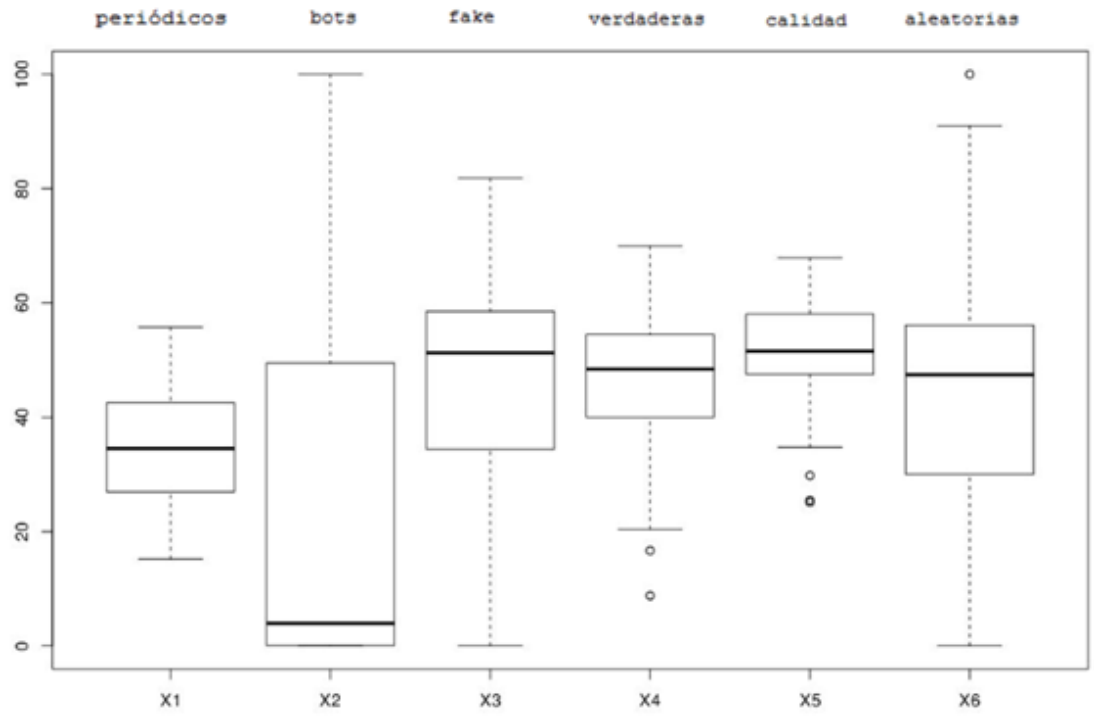
Gracias a la desviación típica podemos conocer qué valores son estándares y cuáles no, es decir, diferenciar los valores que tienen una profundidad grande y otros que son todo lo contrario. Este indicador nos sirve para comparar la variabilidad entre los seis tipos de cuentas de Twitter, y comprobar si la hipótesis de que la distribución de las cuentas de calidad son más heterogéneas que la distribución de las cuentas bots.

Tabla de las medianas del box-plot (valores Q2):

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana	2,6	1,2	3,4	3,5	3,3	3,6

Tabla 6: Medianas del indicador 2

Indicador 3: Porcentaje de tweets que contienen oraciones subordinadas



Si el tweet tiene más de una oración subordinada, solo contabilizamos 1.
Y: Porcentaje de tweets que contienen oraciones subordinadas (nº subordinadas *100/tweets totales)
X: Tipos de cuentas

Tabla 7: Box-plots del indicador 3

Las personas más cultas suelen emplear mayor número de oraciones subordinadas y frases con mayor profundidad, aunque en Twitter no siempre se cumple. Hemos comprobado que las cuentas bots sí utilizan menor profundidad que las demás, y en el anterior box-plot se comprueba que pasa lo mismo entre las oraciones subordinadas. Hay solape prácticamente con todas las cuentas (a excepción de las cuentas de periódicos y calidad), pero si nos fijamos en términos de medianas (Q2) de los box-plot, las cuentas bots tienen un valor de 5, bastante bajo en comparación, por ejemplo, con las cuentas de calidad con valor 53.

Tabla de resultados con las mediadas de cada box-plot del gráfico del porcentaje de subordinadas:

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana (%)	35	5	53	50	53	49

Tabla 8: Medianas del indicador 3

Indicador 4: Número medio de sintagmas adverbiales

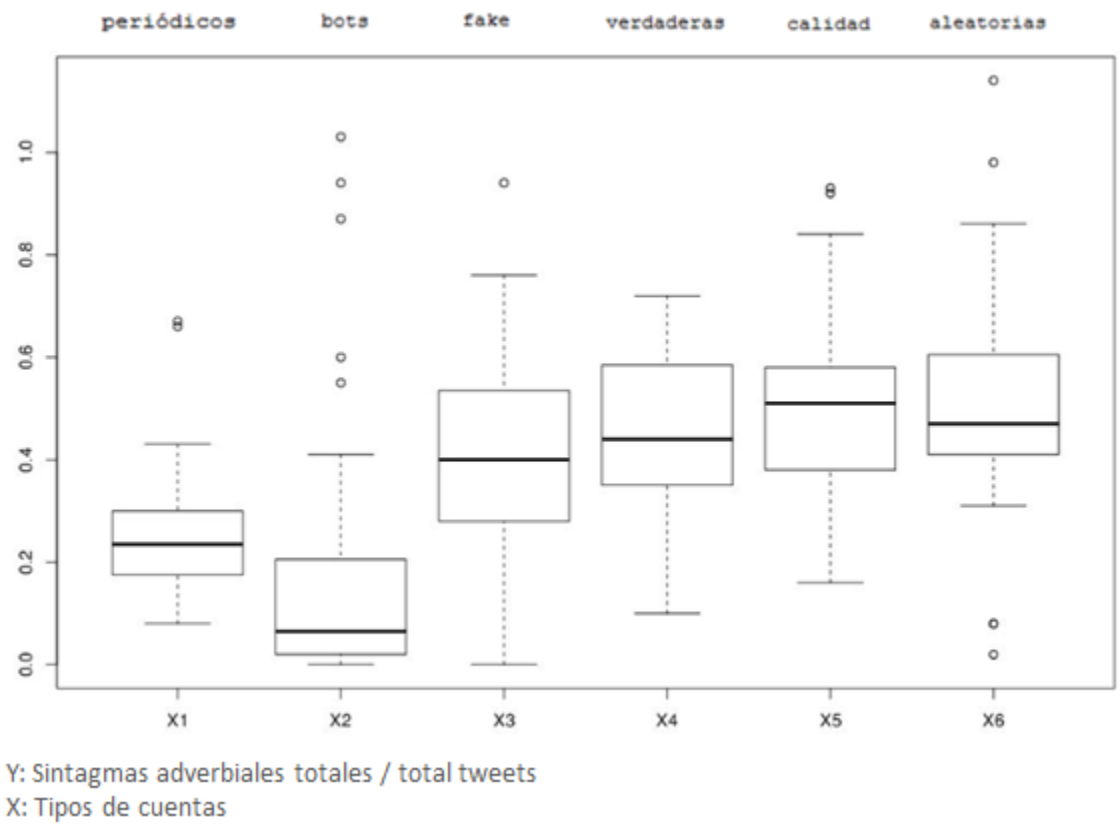


Figura 74: Box-plots del indicador 4

La mediana de los 40 valores de cada tipo de cuenta aparece en la siguiente tabla:

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana	0,25	0,08	0,40	0,46	0,51	0,47

Tabla 9: Medianas del indicador 4

Los bots a parte de tener una profundidad sintáctica menor, carecer de frases subordinadas, menor número de sintagmas adjetivales, también tienen menor número de sintagmas adverbiales. Es coherente que si utilizan menor registro de palabras, va en relación que entre ellas se encuentre adjetivos y adverbios que son los más usados normalmente. No se produce casi solape entre cuentas de periódicos y bots, con cuentas de calidad, por tanto, las cuentas de calidad se pueden distinguir claramente por usar mayor número de sintagmas adverbiales que las otras dos.

El hecho de utilizar un gran número de adverbios como es el caso de las cuentas de calidad, no quiere decir que usen un registro más culto. Siempre dependiendo de qué tipo de adverbios se utilicen, podemos decir si los tweets son más cultos o menos. Adverbios que terminan en mente (en inglés son: rapidly, commonly, normally) son de uso cotidiano y no son acorde con gente que suele utilizar mayor variedad lingüística.

Indicador 5: Número medio de sintagmas adjetivales

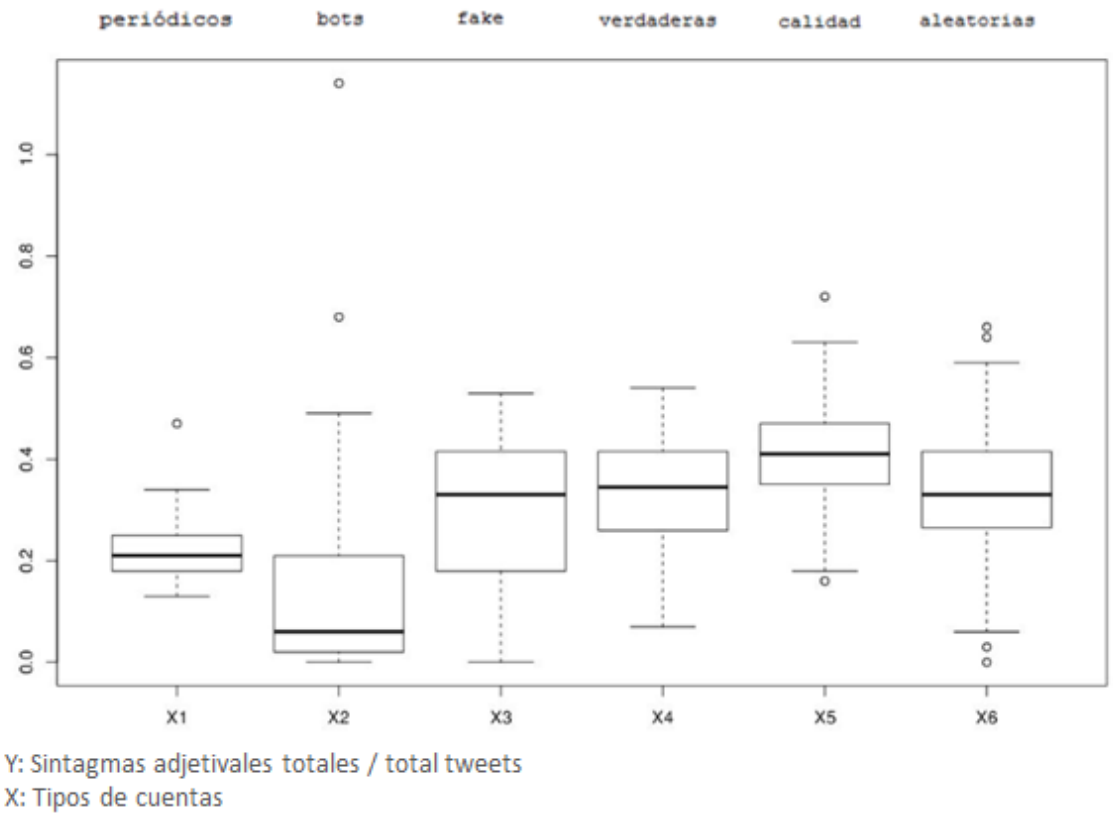


Figura 75: Box-plots del indicador 5

Gracias a la cantidad de sintagmas adjetivales, denotados con la etiqueta ADJP del parser de Stanford, podemos conocer que las cuentas de calidad usan un número mayor, con una mediana de 0.40 frente a una mediana del 0,08 para las cuentas bots, que parece ser que en el contenido de sus tweets no suelen usar este tipo de sintagmas. Con los resultados obtenidos podemos determinar que es un estadístico útil para diferenciar cuentas bots del resto, al tener menor variedad de palabras en general, y cultas en particular, parece ser que los sintagmas adjetivales también se reducen.

La mediana de los 40 valores de cada tipo de cuenta aparece en la siguiente tabla:

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana	0,21	0,08	0,34	0,37	0,40	0,30

Tabla 10: Medianas del indicador 5

Indicador 6: Porcentaje de las últimas 20.000 palabras distintas del Wiktionary que se usan en los tweets

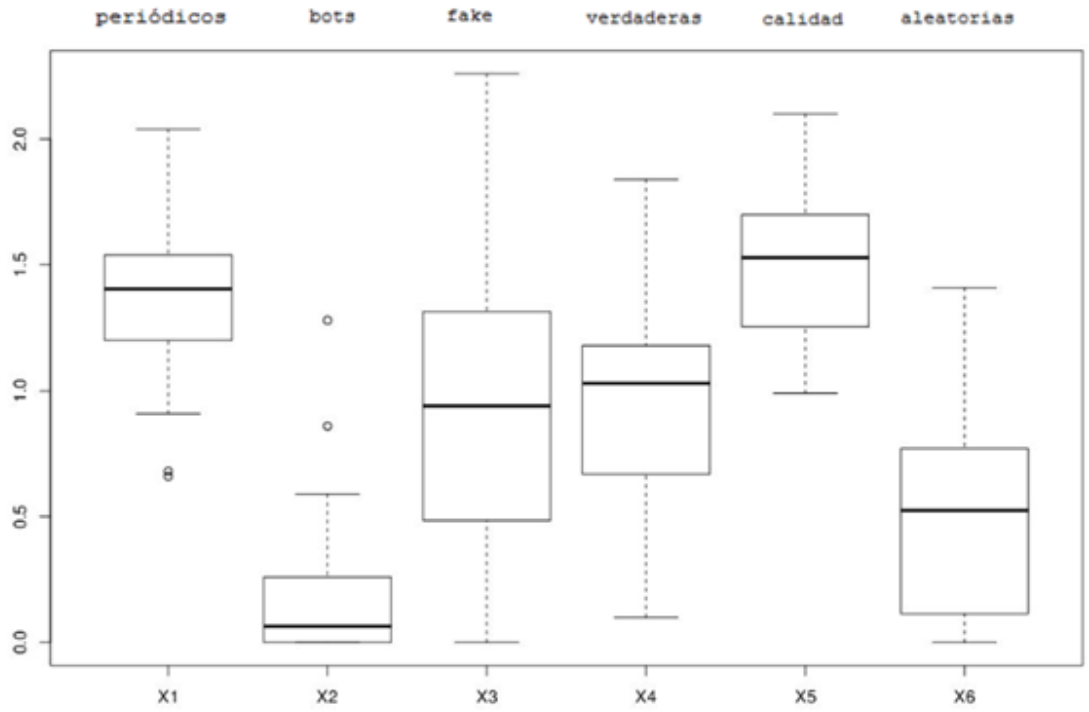


Figura 76: Box-plots del indicador 6

En la imagen anterior, se contempla que las cuentas bots y aleatorias emplean palabras que se usan más habitualmente, por lo que el diagrama de cajas de ambas cuentas se solapan, pero ambas se diferencian del resto. Si analizamos las medianas de los seis box-plot, tenemos:

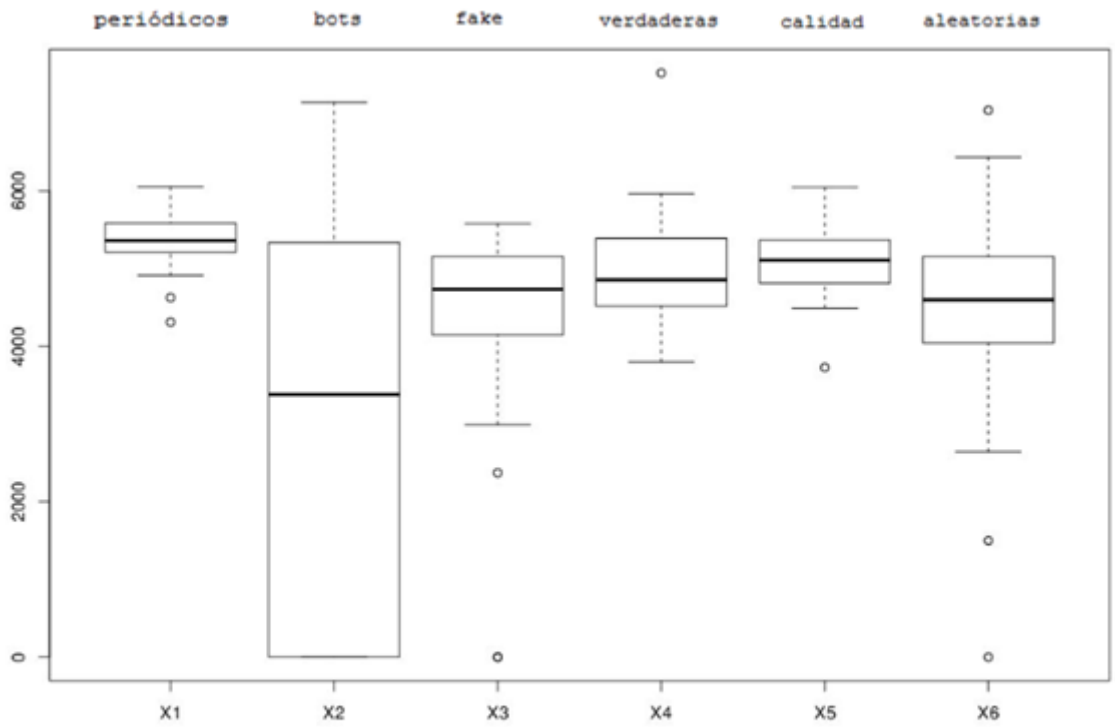
	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana (%)	1,4	0,1	0,9	1,1	1,55	0,55

Tabla 11: Medianas del indicador 6

Las cuentas bots tienen como mediana un 0,1%, lo que quiere decir que el 0,1% representa el valor de la variable de la posición central del conjunto de 40 valores al haber 40 cuentas, frente a un 1,55% de las cuentas de calidad. Claramente las cuentas de calidad utilizan un registro más variado de las últimas 20.000 palabras del Wiktionary con respecto a las cuentas bots. El hecho de usar mayor porcentaje de palabras de las últimas posiciones del Wiktionary frente a los demás, significa que la composición de sus tweets es más rica en cuanto a la variedad de palabras usadas.

Las cuentas de calidad son las que tienen un porcentaje más alto, con una mediana de 1,55 %. Teniendo en cuenta que hay millones de palabras y que es imposible que se registren todas en el Wiktionary, y que los usuarios emplean todas en los 1000 tweets analizados, resulta un porcentaje considerable.

Indicador 7: Desviación típica de la distribución del uso de las últimas 20.000 palabras del Wiktionary



Y: Desviación típica de la distribución del uso de las últimas 20000 palabras del Wiktionary (sin tomar logaritmo)
X: Tipos de cuentas

Figura 77: Box-plots del indicador 7

Observamos en el gráfico que las cuentas de Twitter que son periódicos tienen una desviación típica mayor que el resto de cuentas, y también tienen una mayor media, a pesar de publicar titulares de periódicos con un enlace, muchas de las palabras usadas en esos titulares corresponden a las palabras de las últimas posiciones del Wiktionary, por lo que son de registro culto, y era de esperar al ser periódicos con cuentas oficiales y muy respetadas mundialmente. Los bots también tienen mucha variedad de palabras, abarcan prácticamente todos los valores posibles y es por ello que dificulta poder diferenciar una cuenta bot del resto. Este indicador resulta útil para diferenciar las cuentas de periódicos del resto de cuentas.

5.2. Análisis general de la cuenta de Twitter

Indicador 8: Porcentaje de tweets que son RT

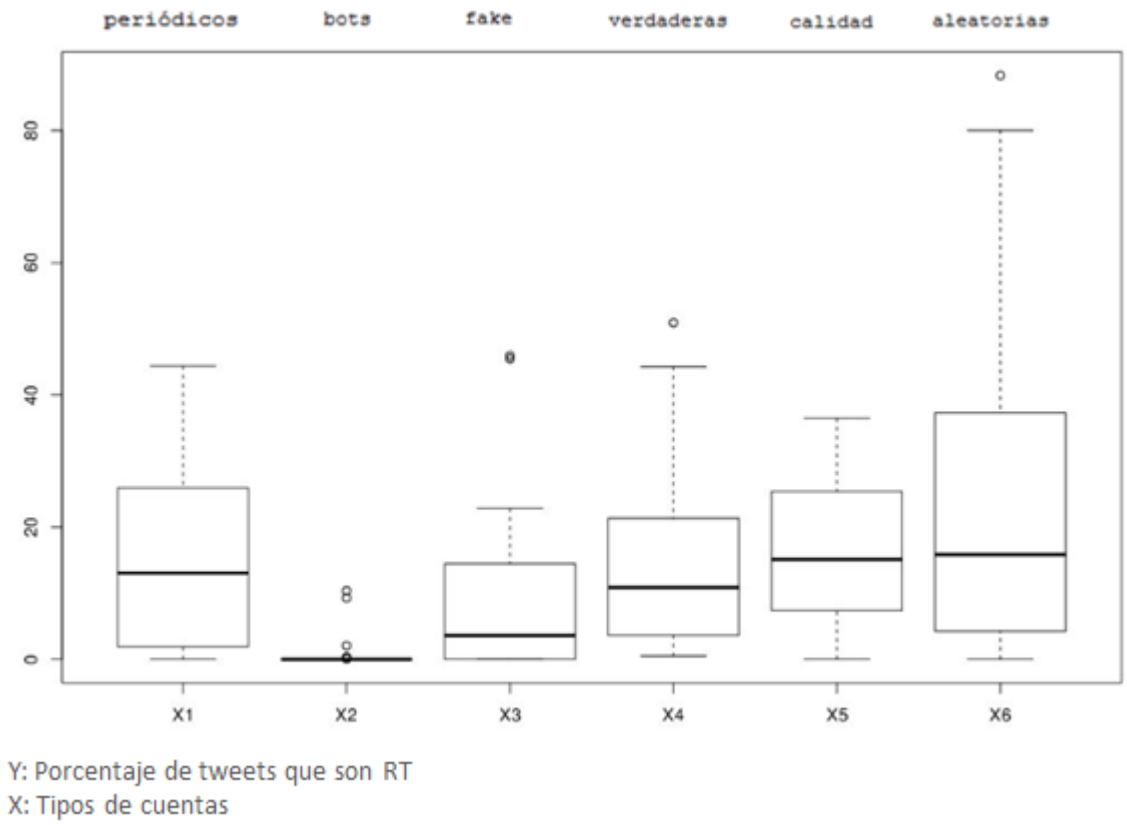


Figura 78: Box-plots del indicador 8

Observamos con la gráfica que las cuentas bots no hacen prácticamente ningún RT ya que no se han escogido aquellos que estén programas para hacer RT a la gente en función de una condición establecida. Las cuentas fake simplemente se dedican a parodiar a cuentas famosas, con sus propios tweets, dejando a un lado los RT. Parece que los usuarios escogidos al azar se dedican más a retuitear a la gente, y con valores más bajos de retuiteo pero también considerables se encuentran los medios de comunicación, por lo que creemos que es para dar a conocer a usuarios que citan en sus periódicos, o al mismo periódico pero con diferentes temáticas distribuidas en distintas cuentas de Twitter, como por ejemplo: economía, política y ciencia. Las cuentas verdaderas tienen una mediana parecida a los periódicos y usuarios escogidos al azar, por lo que lleva a pensar que se hacen RT entre personalidades famosas por interés o simplemente para afianzar una amistad, demostrando que me importa el contenido de tu tweet y lo quiero compartir con el resto de mis seguidores.

Indicador 9: Porcentaje de tweets que contienen menciones

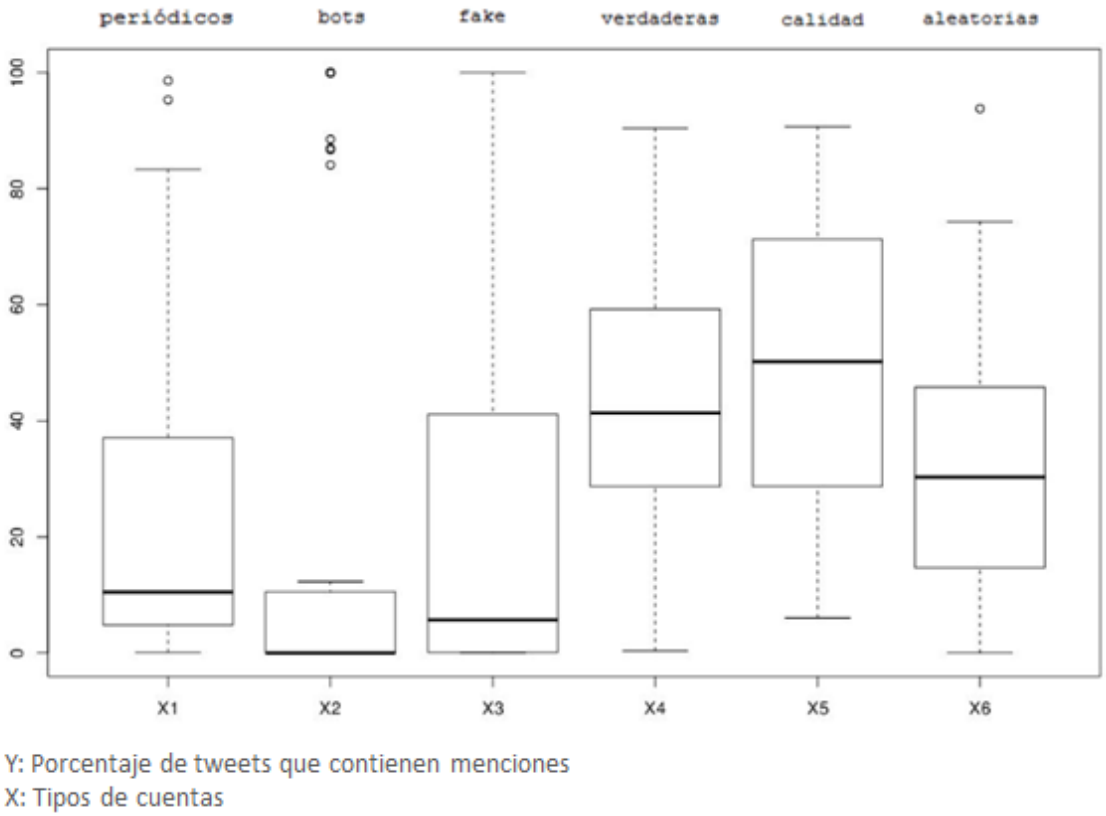


Figura 79: Box-plots del indicador 9

Apreciando los 6 box-plots, se nota que hay solapes en la mayoría de las cajas, exceptuando las cuentas bots, pero hay diferencias significativas a pesar del solape, y que se refieren a la mediana de los valores de cada cuenta, que aparece en la tabla a continuación:

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana (%)	10	0	5	43	50	30

Tabla 12: Medianas del indicador 9

Las cuentas de periódicos normalmente twittean titulares de su periódico y lo acompañan con un enlace, pero no suelen mencionar a gente. Las cuentas bots tienen porcentajes de tweets que contienen menciones que varían del 0% al 12%, siendo el valor de la posición central 0, por lo que es una gran diferencia si lo comparamos con otras cuentas como por ejemplo las de calidad, que tienen la mediana más alta de los seis tipos de cuenta con un valor de 50 %. A parte de la mediana, las cuentas bots presentan solape con los medios de comunicación y cuentas fake, pero se pueden diferenciar de las cuentas verdaderas, de calidad y las aleatorias, según nuestra hipótesis al

visualizar los datos. Se podría decir que los bots elegidos no se dedican a mencionar a usuarios durante el envío de tweets de todo tipo (algunos interesantes y otros serán basura), ya que su fin es transmitir spam masivo sin mencionar a otros usuarios, solo si están programados para dicho objetivo, por eso se piensa que siguen un patrón sintáctico igual en el envío de tweets, tal y como se estudió en el identificador 1 de la sección 5.1.

En las cuentas fake el porcentaje es bastante bajo, ya que se dedican a parodiar a una persona pero sin mencionar ni a ella ni a los demás, incluso muchos se dedican a imitarla, escribiendo tweets como si los escribiera dicha persona, generalmente conocida ya que se trata de una celebridad famosa en muchos casos, aunque uno de sus brazos contiene los porcentajes más altos posibles, lo que lleva a pensar que, en ese caso, si mencionan a la gente que imitan o parodian. Con porcentajes por encima del 30 % se sitúan las cuentas aleatorias, que generalmente twittean mencionando a sus amigos o grupos de música, periódicos, en general, cuentas que siguen y les gusta. Las cuentas verdaderas muchas veces mencionan a sus amigos que suelen pertenecer a un entorno popular, o a series de televisión donde ellos se encuentran trabajando, etc. Las cuentas de calidad son las que tienen un porcentaje mayor que el resto referido a la mediana, ya que son personas del mundo de la ciencia y la cultura, que twittean una idea de investigación o de un libro, recomendación o agradecimiento a gente de su propio trabajo o a sus seguidores, dándole las gracias en forma de agradecimiento por leer su libro o comentar algún aspecto muy concreto y conciso. Las cuentas aleatorias presentan valores altos y bajos, son irregulares, y con respecto al alto porcentaje de menciones creemos que es el resultado de mencionar a sus amigos, como manera de crear vínculos escribiendo a algún seguidor como parte de su tweet.

Indicador 10: Porcentaje de tweets que contienen hashtags

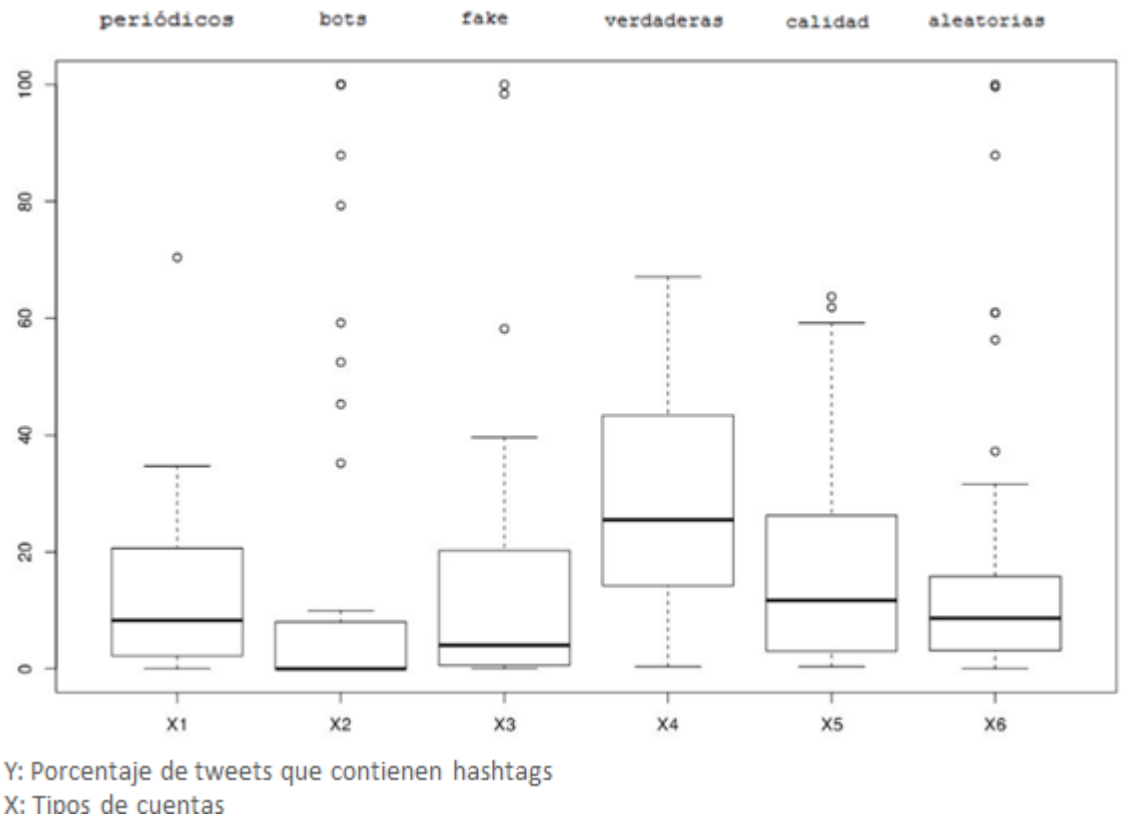


Figura 80: Box-plots del indicador 10

En los box-plots se observa que las cuentas verdaderas son las que twittean con mayor número de hashtags, a pesar de haber solape con las demás, si nos fijamos en las medianas tenemos lo siguiente:

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana (%)	9	0	5	25	11	10

Tabla 13: Medianas del indicador 10

Hay una diferencia significativa entre las cuentas verdaderas y el resto, sobre todo si comparamos su mediana con la mediana de las cuentas bots que es 0. Los utilizan para promover su contenido o el de otros, una buena forma de agrupar a seguidores y que sigan sus partidos, vean sus series, películas o cualquier actividad relacionada. Mientras que las cuentas bots elegidas no suelen twittear con hashtags, sino más bien con enlaces que contienen spam o enlazan a una página con contenido que quieren vender en caso de promocionar un producto de la página online de una tienda. En las cuentas de calidad y medios de comunicación se observa que utilizan los hashtags

para transmitir sus ideas, hacer llegar noticias actuales con una palabra breve que cree interés y que tenga tirón mediático.

Indicador 11: Porcentaje de tweets que contienen enlaces

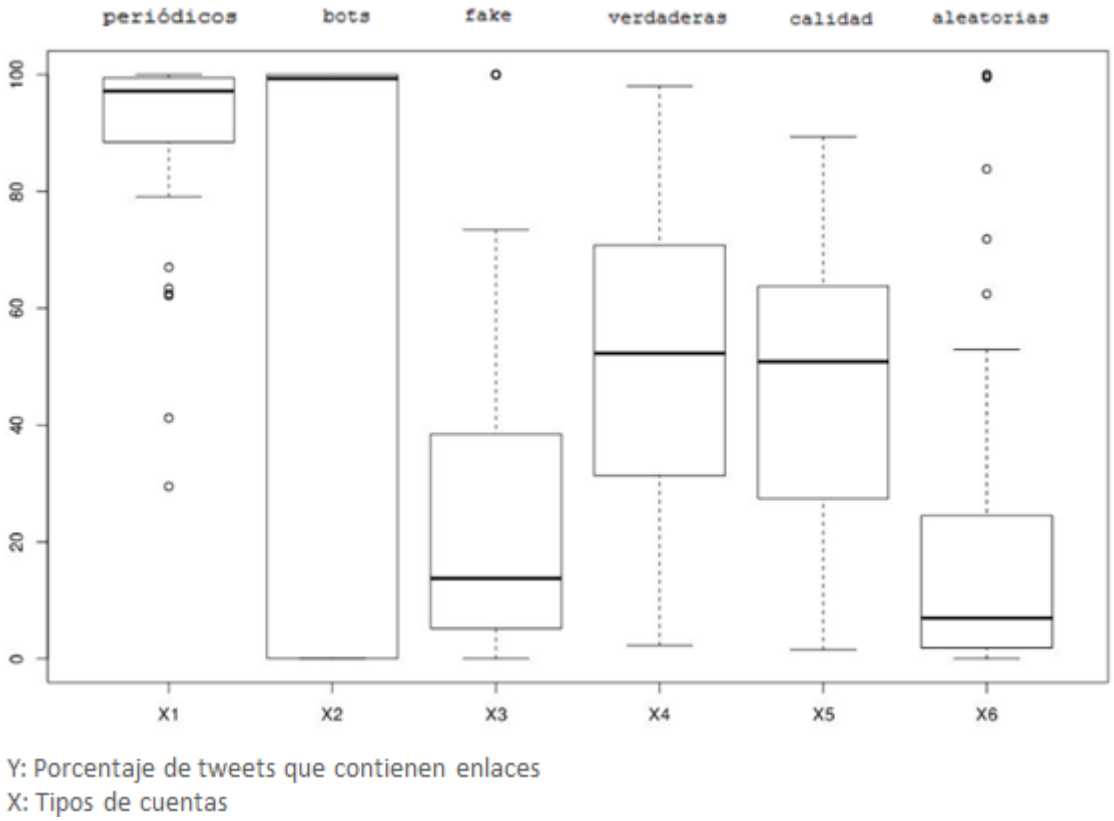


Figura 81: Box-plots del indicador 11

Observamos con el gráfico que los tweets de los periódicos generalmente contienen enlaces, con un margen entre 90% y 98%, es decir, más de 900 tweets sobre 1000 contienen enlaces. El box-plot de las cuentas bots varía entre 0 y 100, con una mediana de 100 %, lo que da a entender que unos bots twittean con enlaces y otros simplemente no están programados para incluir link[32].

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana (%)	98	100	17	50	48	10

Tabla 14: Medianas del indicador 11

[32] Es una referencia a una página web o un contenido específico de un sitio web.

En menor proporción que las cuentas bots y de periódicos se encuentran las verdaderas, creemos que para promocionar algún vídeo o página web y que sus seguidores puedan acceder al recurso. Las cuentas de calidad, con una mediana de 48 %, pensamos que destacan enlaces hacia revistas de investigación con temas publicados en la actualidad, enlaces a universidades, temarios, másters, de forma que puedan atraer a un público intelectual a sus conferencias, seminarios de aspecto tecnológico. Por último, las cuentas simuladas parece ser que se dedican a parodiar a los demás sin utilizar enlaces externos, y las aleatorias tienen un 10 % de mediana, por lo que parece ser que no se centran en escribir enlaces.

Indicador 12: Porcentaje de tweets que son conversación

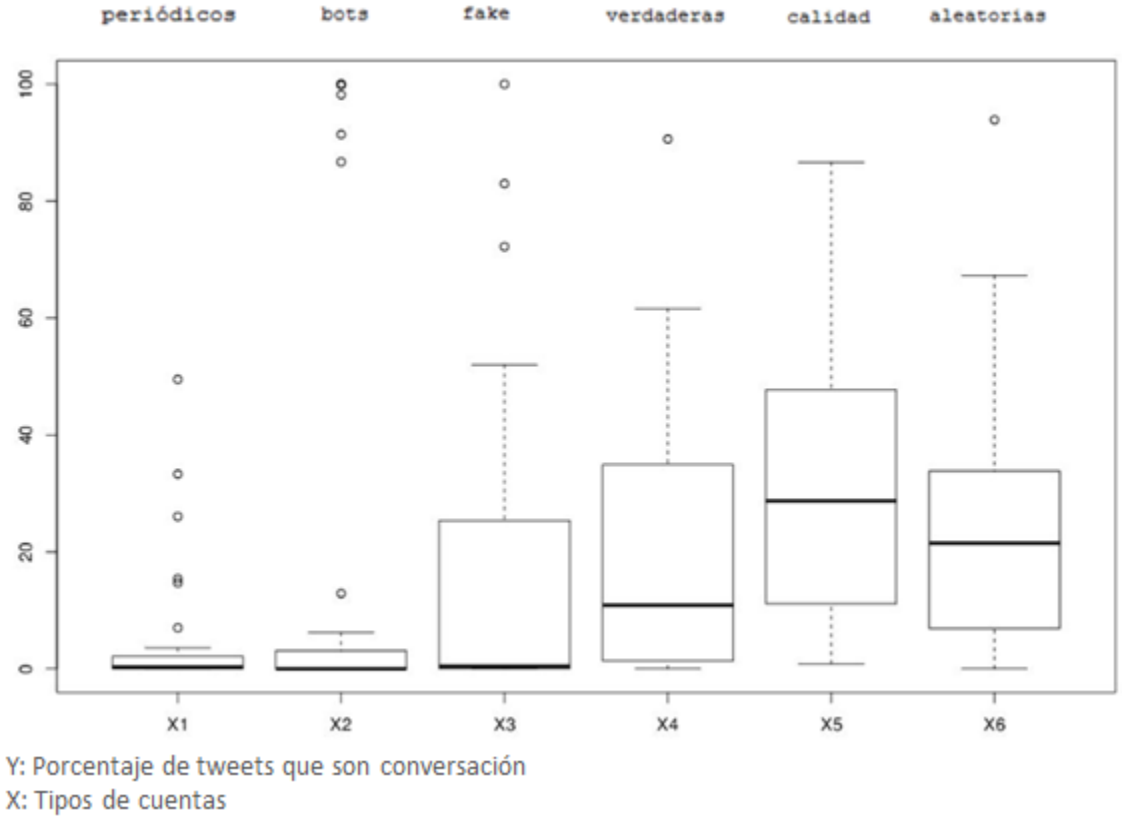


Figura 82: Box-plots del indicador 12

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana (%)	0	0	0	10	30	21

Tabla 15: Medianas del indicador 12

Con la representación gráfica de los box-plots se observa que las cuentas de periódicos se limitan a publicar títulares de sus periódicos, enlaces a artículos de opinión y demás. Las cuentas bots elegidas no suelen conversar, pero si hubiéramos escogido los bots conversacionales tendríamos resultados completamente diferentes, ya que pueden establecer una conversación aleatoria con cuentas que twitteen alguna palabra que dicho programa informático esté programado para responder, comentar y sugerir. También te suelen dar información sobre tus tweets favoritos, número de seguidores, el tiempo, sin que le hayas preguntado por nada. Las cuentas simuladas tienen una mediana de 0, lo que nos lleva a pensar que hacen parodia o imitan a la gente pero sin conversar con el resto, solo leen sus tweets sus seguidores porque se actualiza en su timeline, intercambiando algo de conversación ya que la caja varía de 0-25 %.

Con medianas más altas se encuentran las cuentas de calidad, que parece ser que conversan con personas de su trabajo para comentar los resultados obtenidos en proyectos de universidad, investigación, periódicos y para dar las gracias a la gente que les apoyan y leen sus artículos.

Las cuentas aleatorias que obtuvimos parece ser que tienen un porcentaje alto de tweets conversacionales si los comparamos con el resto, ya que muchas de las 40 cuentas aleatorias pertenecerán a usuarios corrientes que utilizan el Twitter como red social para conversar con sus amigos.

Indicador 13: Número total de usuarios que siguen en Twitter, más conocido como followees

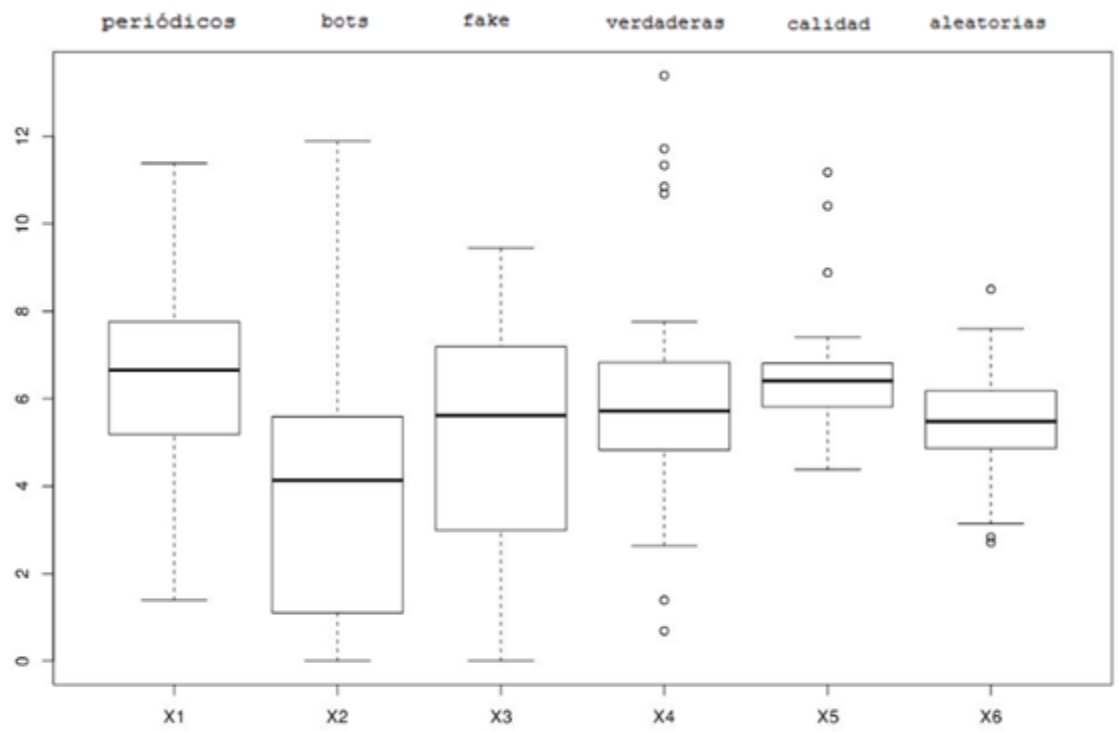


Figura 83: Box-plots del indicador 13

En esta gráfica analizamos el número de personas que siguen en la cuenta de Twitter cada uno de los usuarios escogidos. Según los box-plots, las cuentas bots son las que tienen menor número de followees, por lo que parece ser que no se centran en seguir a los demás en Twitter, sino que twittean publicidad sin descanso, bombardeando a otros usuarios o en su propio Timeline, con criterios de búsqueda de usuarios ya que al no seguir a los demás, deben buscar otras formas de conocer usuarios. También alcanzan valores altos, lo que se demuestra una vez más la irregularidad de dichas cuentas programadas, por la heterogeneidad de los resultados. Va estrechamente relacionado con los seguidores, que se detalla en el identificador 14.

Hemos obtenido valores altos con las cuentas de medios de comunicación, por lo que pensamos que siguen a todos sus miembros del periódico, y a personas que son de interés común para el periódico, ya sean presidentes, ministros, medios de otros países, en definitiva mantienen el contacto con muchas cuentas, y luego veremos si esos followees son correspondidos con followers en relación. En las cuentas de calidad también se ve que suelen seguir a mucha gente, y las verdaderas también, para mantener el contacto con sus amigos o gente importante para ellos, ya se intelectualmente o de otro tipo. Si analizamos las cuentas fake, observamos que también son muy variables, ya que por una parte es lógico que sigan a muchos usuarios para esperar followers y así darse a conocer en

Twitter, para poder leer todos los tweets y así saber cómo tienen que imitar o parodiar a los que deseen. Para los valores más bajos, es posible que imiten a determinados famosos, y como dichos famosos no suelen seguir a mucha gente, pues ellos tampoco para no levantar sospechas.

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana	7,7	4,1	6,6	6,8	7,3	6,3

Tabla 16: Medianas del indicador 13

Indicador 14: Número de seguidores en Twitter

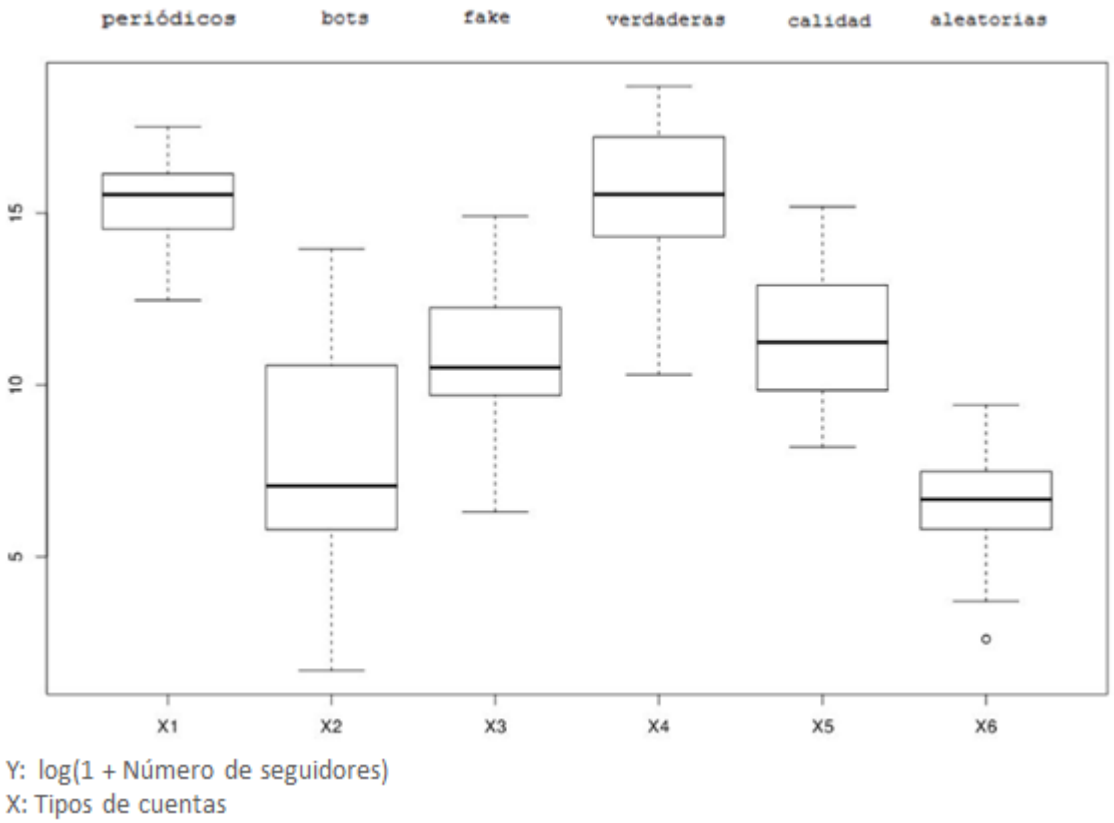


Figura 84: Box-plots del indicador 14

Los resultados obtenidos eran de esperar según nuestras hipótesis. Hemos tomado el logaritmo para mayor claridad en los resultados. Las cuentas de periódicos obtienen una gran cantidad de seguidores en Twitter, ya que cada vez son más las personas que les gusta estar actualizada en cada momento de lo que sucede en todo el mundo.

De cara a las cuentas verdaderas, muchas de ellas son famosas, tanto del mundo deportivo como del cine y televisión. Tienen un número de seguidores muy alto debido a su estatus social, y muchas

de ellas son muy activas en la red social Twitter, por lo que atraen a más gente con sus comentarios y suelen ser trending topic cuando logran un premio, ganan un campeonato, momento televisivo de éxito o simplemente tienen un error.

Con resultados muy parecidos se encuentran las cuentas de personas del mundo de la ciencia y la cultura. Profesores de universidad e investigadores son conocidos en su ámbito laboral, pero a no ser que sea una eminencia, no suelen tener el mismo éxito fuera. De cara a escritores y periodistas, muchos de ellos son conocidos, y más aún si salen en la televisión para dar alguna entrevista televisiva o en algún debate o programa televisivo. Por último se encuentran las cuentas bots y aleatorias con porcentajes parecidos.

Las cuentas aleatorias al ser personas corrientes no suelen ser conocidas y, por ello, suelen tener menos followers. En el tabla 19 aparecen las medianas de este identificador, donde las aleatorias tienen la mediana más baja si la comparamos con las cuentas de periódicos o verdaderas al ser menos conocidas en Twiter.

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana	15,8	7	10,5	15,5	11	6,7

Tabla 17: Medianas del indicador 14

5.3. Análisis temporal

Indicador 15: Intervalo máximo entre un tweet y el siguiente

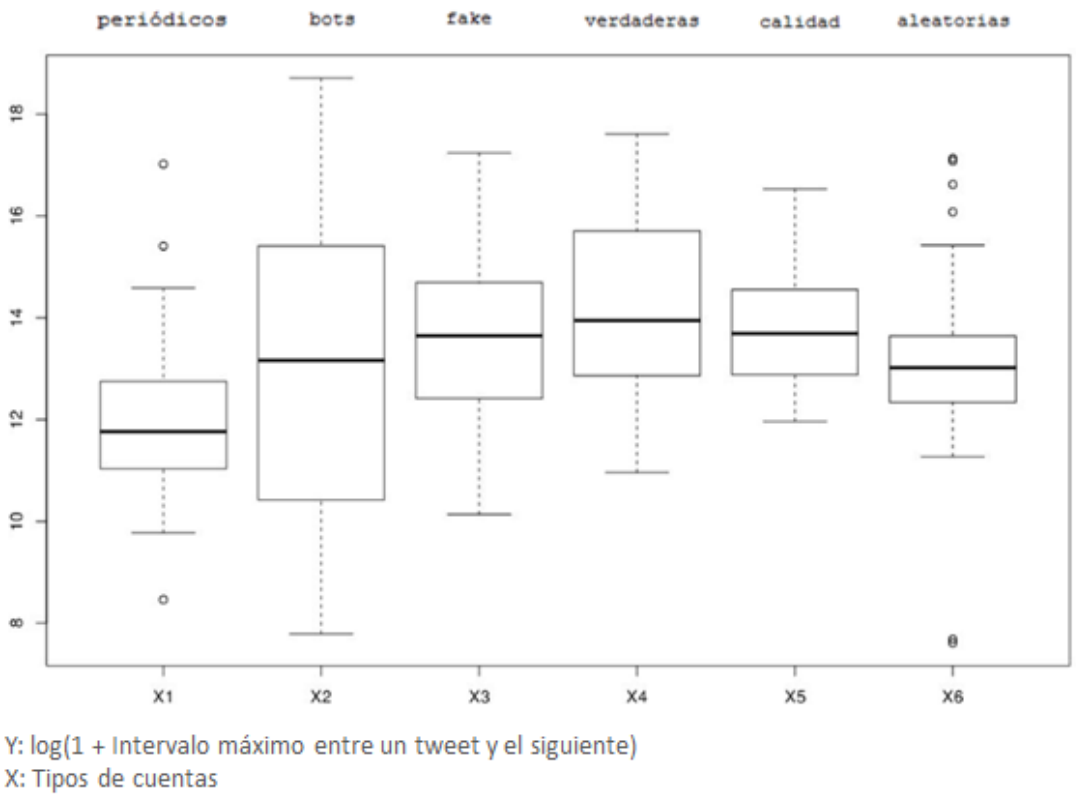


Figura 85: Box-plots del indicador 15

Con este indicador pretendemos averiguar cuánto tiempo transcurre entre un tweet y su anterior, calculamos todas las diferencias posibles y nos quedamos con el máximo valor de todos ellos. Normalmente las personas postean una cantidad de tweets considerables en un rato, que es cuando se conectan a Twitter o tienen un rato libre, y luego suelen dejar pasar bastante tiempo ya que tienen que ir a trabajar, hacer la comida, o dormir. Hay momentos del día que son más factibles para twittear tranquilamente sobre una noticia que quieres compartir o leer de tu timeline y contestar a tus seguidores. Sin embargo, si lo comparamos con programas informáticos que twittean a todas horas sin diferenciar la noche del día, horas de comer y de descanso, pues resulta un dato importante a investigar y poder diferenciar este tipo de cuentas en Twitter. Según los box-plots, las cuentas de periódicos tienen los intervalos máximos entre dos tweets consecutivos menores que el resto de tipos de cuentas. Esto es así porque dichas cuentas son como los bots que tuitean noticias o artículos en blogs. Son automáticas pero basadas en titulares generados por humanos, y tuitean la noticia en el momento que se produce, por lo que el intervalo entre dos tweets consecutivos es muy pequeño, y por tanto, el máximo intervalo de entre todos ellos será menor en comparación con otras cuentas como pueden ser las cuentas fakes, las verdaderas y las cuentas de calidad. En las cuentas bots observamos que la mediana de los 40 valores resulta algo más baja que las fake, verdaderas y

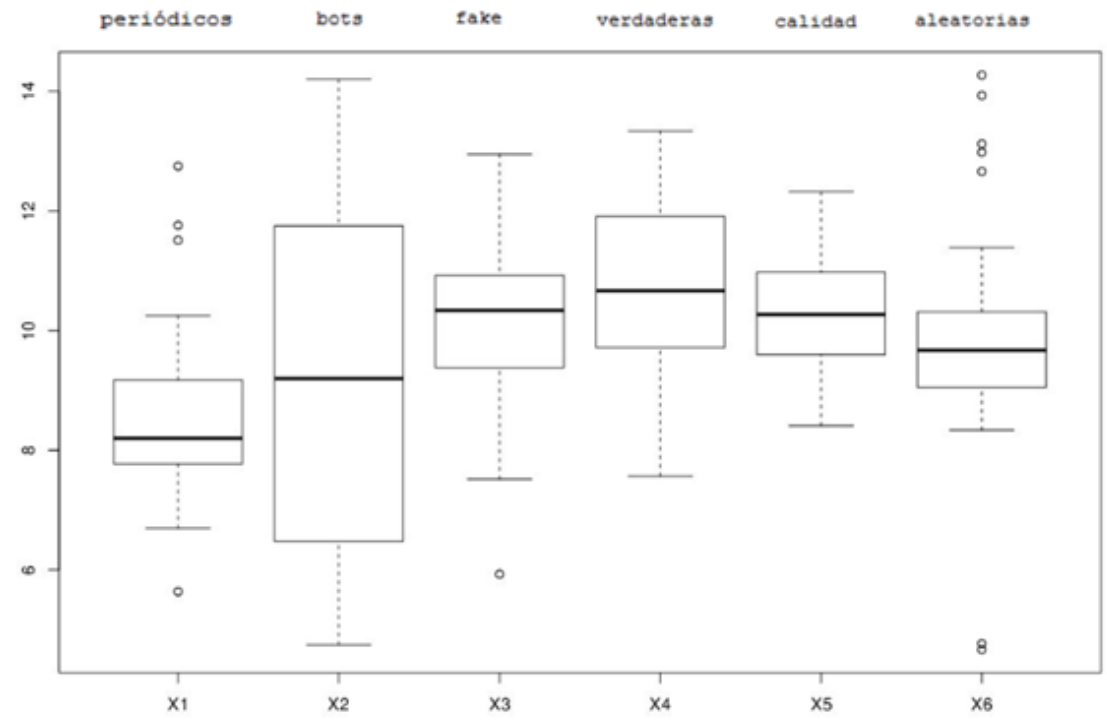
calidad, pero tiene mucha variedad, desde valores entre 10,5 y 14,6 (hablando de valores logarítmicos). No es nada regular. Las cuentas aleatorias tienen valores bajos, son cuentas de todo tipo, y según los resultados observamos que la mayoría de ellas son activas en Twitter, con frecuencias de twitteo muy altas, por lo que al tener una regularidad de tweets, el tiempo de separación entre ellos se reduce y el máximo valor entre dos tweets consecutivos es menor en comparación a los demás, igualando a las cuentas bots y solo tienen tiempos menores las cuentas de periódicos.

La tabla con las medianas corroboran lo explicado anteriormente:

	Periódicos	Bots	Fake	Verdaderas	Calidad	Aleatorias
Mediana	11,8	13	13,7	13,9	13,8	12,9

Tabla 18: Medianas del indicador 15

Indicador 16: Desviación típica de la diferencia de segundos entre un tweet y el siguiente



Y: log(1 + Desviación típica de la diferencia de segundos entre cada tweet y su siguiente)
X: Tipos de cuentas

Figura 86: Box-plots del indicador 16

Se aprecia en los box-plots que las cuentas de periódicos tienen una dispersión de twitteo más baja en relación con las cuentas fake, verdaderas, calidad y aleatorias, lo cual era de esperar ya que son programas informáticos que están programados para postear noticias en tiempo real, para mantener informados a todos su seguidores lo más rápido posible. En cuanto a las cuentas bots, son las cuentas que tienen menores desviaciones típicas de la diferencia de segundos entre cada tweet y su anterior. Hemos tomado el logaritmo natural a cada desviación típica para poder diferenciar los 6 tipos de cuentas al tener valores más bajos. Entre los valores 0 y 6.5 se encuentran resultados de la desviación típica de las cuentas bots, que no solapan con los demás a excepción de dos valores atípicos. Se intenta comprobar la hipótesis de que las cuentas bots, al ser programas informáticos programados para twittear en todo momento sin descanso alguno, tendrán las diferencias de tiempos menores entre cada dos tweets, y se podrá diferenciar con las cuentas humanas, que deberían descansar en ciertas horas del día como personas humanas que son.

6. Conclusiones y Trabajos futuros

En este capítulo se analiza los resultados obtenidos del proyecto de investigación, presentando las principales conclusiones alcanzadas tras la evaluación del mismo. A continuación se esbozan también algunos puntos mejorables sobre los que se podría ampliar el trabajo ya realizado en este proyecto.

6.1. Conclusiones

En esta sección se van a exponer las conclusiones que se han extraído tras la finalización del trabajo desarrollado, comentando los resultados obtenidos, las dificultades que se han tenido durante todo el desarrollo del proyecto y las conclusiones personales.

Una vez concluido el trabajo de investigación, se puede afirmar que se ha logrado el objetivo principal del proyecto, dotando al usuario final de una plataforma capaz de capturar tweets en la red social Twitter, calcular una serie de indicadores en base a esos mensajes, detectar la calidad de un usuario en Twitter y poder distinguir unos tipos de cuenta de otras.

Se pueden implementar nuevos indicadores para completar cualquier estudio sobre la calidad de usuarios en función de sus mensajes en Twitter, por lo que este proyecto se pude reutilizar para cualquier investigación relativa al procesamiento del lenguaje natural y usando la red social Twitter.

El hecho de utilizar software libre y gratuito posibilita a otras personas a hacer uso de esta plataforma sin coste alguno, y dado que he proporcionado todo el código fuente a la Universidad, esto hará que otros compañeros puedan continuar con esta línea de investigación tan interesante.

En cuanto a los resultados obtenidos, se han sacado las siguientes conclusiones:

- A través de los indicadores analizados hemos comprobado que se pueden diferenciar cuentas bots (excluyendo las que se dedican a retuitear únicamente, tuitear titulares de periódicos, y aquellas que son conversacionales), con cuentas de personas humanas.
- Los resultados demuestran que las cuentas bots twittean siguiendo patrones sintácticos iguales al estar programados para postear una cantidad abusiva de tweets con una pequeña

modificación en su contenido. La frecuencia de sintagmas adjetivales, adverbiales y oraciones subordinadas es escasa, y la profundidad del árbol sintáctico es menor en comparación con el resto, al tratarse de indicadores propios de cuentas más cultas. Se preocupan poco por su calidad lingüística de sus tweets, donde también se refleja en la poca variedad de palabras más elaboradas que suelen utilizar en sus mensajes. Con datos como el gran porcentaje de tweets que contienen enlaces, nos ayuda a diferenciar bots de humanos ya que se centran en enviar publicidad en Twitter, a veces empleando enlaces maliciosos y les importa poco el contenido de los demás e interactuar con la gente (a no ser que sean conversacionales que simulan mantener una conversación con una persona), al igual que se ha corroborado que el tiempo máximo entre un tweet y su anterior es menor en bots, al inundar Twitter con tweets constantemente y no dejando descanso alguno salvo cuando alcanzan el límite impuesto por Twitter.

- Otra utilidad de esta aplicación es la diferenciación de determinadas cuentas que son escritas por personas humanas. Las cuentas de medios de comunicación y las de calidad utilizan mayor variedad de palabras más instruidas que cualquiera de las demás, que era de esperar al tratarse de personas con un nivel cultural por encima de las demás. Las personas famosas contienen un alto porcentaje de seguidores en Twitter, pero también lo tienen los periódicos, lo que nos lleva a pensar que, a pesar de la revolución que tienen algunas personalidades famosas en el mundo del Twitter, no se deja a un lado los periódicos y nos gusta estar al corriente de lo que sucede en todo el mundo.
- Las cuentas fake son difíciles de detectar y no hemos encontrado identificadores suficientes para poder diferenciarlas con el resto de cuentas. Sin embargo, las cuentas aleatorias son distinguibles por su escasez de seguidores, y poca variedad de palabras cultas en sus tweets, que concuerda con el pensamiento de que son usuarios corrientes, que utilizan Twitter como medio de interactuar con otras personas sin entrar en detalle en el contenido de sus tweets.

Para finalizar, como conclusión personal, este proyecto me ha aportado mucho, ya que he aprendido en detalle la red social Twitter que está en auge actualmente, y es bastante interesante para la realización de multitud de estudios. He aprendido a trabajar con una metodología ágil y el haber empezado a trabajar en el Departamento de Ingeniería Telemática me ha hecho crecer como profesional, y trabajar al igual que si estuviera en una empresa. Considero que este trabajo me ha abierto muchas puertas, y se ha demostrado al explicar mi trabajo de investigación en las diferentes entrevistas que he podido realizar desde que me asignaron este proyecto.

6.2. Líneas futuras

El trabajo efectuado ha dejado líneas abiertas para posibles investigaciones futuras. A continuación proponemos las que nos parecen más importantes.

- La posibilidad de analizar los tweets en diferentes lenguajes de programación, usando las librerías específicas para cada caso. Al igual que el análisis de tweets en más idiomas, no solo el inglés.
- Detectar la calidad de otros tipos de cuentas de Twitter, como por ejemplo, estudiar diferentes cuentas bots, cada una de ellas con su finalidad correspondiente, ya sea tuitear titulares, conversar con el resto de usuarios de twitter.
- Permitir analizar los tweets en determinadas franjas horarias del día, buscando aquellas cuentas que permiten publicar la ubicación de sus tweets, para poder hacer nosotros la conversión horaria correspondiente.
- Añadir más tipos de identificadores, siempre buscando la manera de identificar la calidad de un usuario en Twitter de todas las formas posibles. Se podría analizar las urls maliciosas y las IP[33] de los ordenadores que postearon los tweets, y comprobar si aparecen en la página web dedicada a actualizarlas lo antes posible, aunque siempre se irá con retraso al crecer el número de enlaces nuevos, y será difícil encontrar dichas IPs al poder esconder nuestra IP bajo una falsa o simplemente cambiándola.
- Migrar el proyecto a Android, de modo que sea más práctico.

[33] Dirección de protocolo de Internet.

Referencias

[1] «PLOS ONE: Scaling-Laws of Human Broadcast Communication Enable Distinction between Human, Corporate and Robot Twitter Users». [En línea]. Disponible en: http://www.plosone.org/article/info%253Adoi%252F10.1371%252Fjournal.pone.0065774. [Accedido: 03-mar-2014].

[2] «@spam: the underground on 140 characters or less - p27-grier.pdf». .

[3] «2010 ACSAC Proceedings - Frontmatter - acsac10.pdf». .

[4] «The Stanford NLP (Natural Language Processing) Group». [En línea]. Disponible en: http://nlp.stanford.edu/software/lex-parser.shtml. [Accedido: 03-mar-2014].

[5] «dependencies_manual.pdf». .

[6] «pcfgs.dvi - pcfgs.pdf». .

[7] «The University of Pennsylvania (Penn) Treebank Tag-set». [En línea]. Disponible en: http://www.comp.leeds.ac.uk/ccalas/tagsets/upenn.html. [Accedido: 03-mar-2014].

[8] «Downloads - language-detection - Language Detection Library for Java - Google Project Hosting». [En línea]. Disponible en: https://code.google.com/p/language-detection/downloads/list. [Accedido: 03-mar-2014].

[9] «Changing Bits: Accuracy and performance of Google's Compact Language Detector». [En línea]. Disponible en: http://blog.mikemccandless.com/2011/10/accuracy-and-performance-of-googles.html. [Accedido: 11-mar-2014].

[10] «Centro de Ayuda de Twitter | Sobre los Límites de Twitter (Actualizaciones, API, MD y Seguimiento)». [En línea]. Disponible en: https://support.twitter.com/articles/344781-sobre-los-limites-de-twitter-actualizaciones-api-md-y-seguimiento#. [Accedido: 04-mar-2014].

[11] «REST API Rate Limiting in v1.1 | Twitter Developers». [En línea]. Disponible en: https://dev.twitter.com/docs/rate-limiting/1.1. [Accedido: 25-mar-2014].

[12] «OAuth FAQ | Twitter Developers». [En línea]. Disponible en: https://dev.twitter.com/docs/auth/oauth/faq. [Accedido: 13-mar-2014].

[13] «ElProtocoloOAuth - keejoo - Explicación del protocolo OAuth - Keejoo. Un cliente twitter escrito en GTK3 - Google Project Hosting». [En línea]. Disponible en: https://code.google.com/p/keejoo/wiki/ElProtocoloOAuth. [Accedido: 13-mar-2014].

[14] «Java (lenguaje de programación) - Wikipedia, la enciclopedia libre». [En línea]. Disponible en: http://es.wikipedia.org/wiki/Java_%28lenguaje_de_programaci%C3%B3n%29. [Accedido: 03-mar-2014].

[15] «Programación en Java/Características del lenguaje - Wikilibros». [En línea]. Disponible en: http://es.wikibooks.org/wiki/Programaci%C3%B3n_en_Java/Caracter%C3%ADsticas_del_lenguaje. [Accedido: 03-mar-2014].

[16] «Eclipse y Java - TutorialEclipse.pdf». .

[17] «Eclipse, entorno de desarrollo integrado - EcuRed». [En línea]. Disponible en: http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado. [Accedido: 03-mar-2014].

[18] «Twitter4J - A Java library for the Twitter API». [En línea]. Disponible en: http://twitter4j.org/en/index.html#download. [Accedido: 03-mar-2014].

[19] «Wiktionary:Frequency lists - Wiktionary». .

[20] «Wiktionary:Frequency lists - Wiktionary». [En línea]. Disponible en: http://en.wiktionary.org/wiki/Wiktionary:Frequency_lists#English. [Accedido: 13-mar-2014].

[21] «Diagrama de caja - Wikipedia, la enciclopedia libre». [En línea]. Disponible en: http://es.wikipedia.org/wiki/Diagrama_de_caja. [Accedido: 04-mar-2014].
[22] «Introduccion al entorno R.pdf». .
[23] «Twitter / Política de Privacidad de Twitter». [En línea]. Disponible en: https://twitter.com/privacy. [Accedido: 03-mar-2014].
[24] «¿Que es la Lopd?» [En línea]. Disponible en: http://cuidatusdatos.com/infolopd.html. [Accedido: 12-mar-2014].
[25] «La API de Twitter y la Ley Orgánica de Protección de Datos (LOPD)». [En línea]. Disponible en: http://estwitter.com/2010/07/02/la-api-de-twitter-y-la-ley-organica-de-proteccion-de-datos-lopd/. [Accedido: 12-mar-2014].
[26] «The 100 most influential news media Twitter accounts | memeburn». [En línea]. Disponible en: http://memeburn.com/2010/09/the-100-most-influential-news-media-twitter-accounts/. [Accedido: 19-mar-2014].
[27] «The List: Ten fake Twitter accounts - FT.com». [En línea]. Disponible en: http://www.ft.com/intl/cms/s/2/c843804c-3b21-11e2-b3f0-00144feabdc0.html#axzz2jz08haZW. [Accedido: 19-mar-2014].
[28] «15 Fake and Funny Twitter Accounts | PCWorld». [En línea]. Disponible en: http://www.pcworld.com/article/159492/fake_funny_twitter.html. [Accedido: 19-mar-2014].
[29] «Top 15 Best Fake Twitter Accounts of 2012 | HEAVY». [En línea]. Disponible en: http://www.heavy.com/comedy/2012/12/the-15-best-fake-twitter-accounts-of-2012/. [Accedido: 19-mar-2014].
[30] «Check out 8 first class Twitter bots. | Digital Trends». [En línea]. Disponible en: http://www.digitaltrends.com/social-media/the-10-best-twitter-bots-you-arent-following/. [Accedido: 03-mar-2014].
[31] «Java: Manejo y formateo de Fechas DateFormat y SimpleDateFormat | Sur Patterns». [En línea]. Disponible en: http://surpatterns.com/sitio/tutoriales-programacion-surpatterns/java-manejo-y-formateo-de-fechas-con-dateformat-y-simpledateformat/. [Accedido: 04-mar-2014].

Apéndices

A. Presupuesto

En este apartado se realiza el desarrollo del Trabajo Fin de Grado en términos económicos. Se presenta una planificación que contempla todas las tareas llevadas a cabo durante el desarrollo del proyecto, incluyendo las tareas críticas, un análisis detallado del personal que ha realizado el proyecto, las horas empleadas, y la inversión necesaria en equipos y software.

A.1 Tareas

Se procede a comentar las diferentes etapas en las que se ha ido realizando el proyecto, habiendo realizado pruebas periódicas y actualizando el software actual.

Se comenzó en el mes de junio de 2013 y finalizando a finales de marzo de 2014, un total de 10 meses aproximadamente, que traducido en jornadas de trabajo de ocho horas diarias equivaldrían a mil seiscientas horas diarias, Desde el 1 de diciembre del 2013, lo realizo en el Departamento de Ingeniería Telemática en la Universidad Carlos III de Madrid.

Fases y problemas encontrados durante el desarrollo del proyecto:

- **Fase 1**

  Estudio en profundidad del parser de la Universidad de Stanford, centrándonos en analizar sintácticamente frases de ejemplo y averiguando cómo podíamos conocer si eran subordinadas, encontrar determinados sintagmas y saber la profundidad del árbol sintáctico. También realicé una presentación a algunos profesores del departamento de telemática sobre dicha herramienta.

- **Fase 2**

  Dedicada a la detección del idioma del texto leído de un fichero, entendiendo en detalle todas las clases que vienen implementadas en el paquete descargado, y creando los métodos necesarios para poder determinar si una frase está escrita en inglés.

- **Fase 3**

  Una de las partes más pesadas fue la búsqueda de todas las cuentas de usuarios en Twitter que queremos analizar, unas eran más fáciles ya que venían en páginas oficiales

como los periódicos, y otras eran más difíciles de encontrar como las cuentas bots, fake, las verdaderas y las de calidad, donde se realizó una búsqueda exhaustiva en Twitter. Las cuentas de calidad Las cuentas aleatorias utilizando el Streaming API se obtenían al instante.

- **Fase 4**

  Se llevó a cabo el aprendizaje en la utilización de una librería para el API de Twitter en lenguaje Java, estudiando la manera de obtener los tweets de cada usuario, conociendo los límites impuestos por la API, y tratar de averiguar cómo saber si un tweet es RT, es conversación, o si contiene hashtags, enlaces y meciones.

- **Fase 5**

  Primeramente nos dedicamos a establecer una estructura del proyecto, con el número de clases a desarrollar y los módulos principales. Una de las partes más laboriosas de conseguir fue la obtención del número de patrones sintácticos, al tener que separar el etiquetado de las palabras del tweet. La parte de analizar las palabras del tweet que aparecían en el Wiktionary también llevó bastante trabajo, y se dedicaron muchas horas. Desarrollamos el código para calcular el porcentaje de tweets posteados en diferentes franjas horarias del día, pero no tuvimos en cuenta que el API de Twitter no nos proporciona la hora local en que se posteó cada tweet, solo la hora a la que le llegó a Twitter, por lo que uno de los problemas presentados. El trabajo relacionado citado en la sección 2.3.1 también realizó este estudio, pero ellos solo consideraron las cuentas de usuarios de Twitter que habían decidido mostrar la ubicación de sus tweets, por lo que fácilmente se puede hacer la conversión horaria. Dada la complejidad en la búsqueda de cuentas bots y fake, y además, que twittearan en inglés y tuvieran un número de tweets considerables para su análisis, si le añadimos la comprobación de la ubicación de sus tweets, tendríamos que bajar de las 40 cuentas, e incluso de 30, y los resultados no serían estadísticamente significativos.

- **Fase 6**

  Se realizaron multitud de pruebas en la generación de los box-plots para representar los valores de cada identificador, tomando logaritmo natural en algunos casos para mayor claridad de los gráficos, esperando 12 horas por cada ejecución y cambiando de un sistema operativo a otro según la tarea.

- **Fase 7**

Conclusiones obtenidas con el análisis de cada identificador, centrándonos en que no haya solape con los box-plots y ver cuales nos sirven para diferenciar unas cuentas de otras. El hecho de realizar continuamente pruebas para generar los identificadores correctamente, hace que las conclusiones varíen hasta que, finalmente, se obtiene el proyecto que se planteó en un principio.

- **Fase 8**

  Esta última parte requiere concentración, y mucho tiempo. A pesar de que a lo largo del trabajo hemos ido apuntando las referencias usadas mediante el programa zotero, comentando el código elaborado y toda la información útil en el momento del desarrollo, al final quedan muchos puntos que completar y requiere un esfuerzo adicional considerable redactar la memoria final.

Tabla que muestra el número de horas empleadas por el personal del proyecto para atender las diferentes partes del proyecto:

Fases	Mes	Labor	Horas
1	Junio/Julio	Estudio del parser de Stanford	160
2	Agosto	Manejo del proyecto language detection de google	80
3	Septiembre	Búsqueda de los 6 tipos de cuentas	200
4	Octubre	Manejo del API de Twitter	200
5	Octubre/Noviembre	Diseño e implementación del programa principal	320
6	Diciembre	Pruebas	160
7	Enero	Conclusiones	160
8	Febrero/Marzo	Documentación del proyecto	320
	Horas totales		**1.600**

Tabla 19: Fases, labores y horas realizadas durante el proyecto

Reuniones

Al permanecer trabajando en el departamento de telemática, nos visitábamos frecuentemente y manteníamos reuniones para comprobar el trabajo realizado, y las posibles mejoras que se podían establecer.

A.2 Análisis de costes

Detallamos los costes asociados al proyecto, separados en costes de personal y costes de materiales.

A.2.1 Costes de personal

El personal dedicado al desarrollo de este proyecto se compone de un tutor llamado Luis Sánchez Fernández, y del autor Tello Ismael Miñana Rontomé.

Personal	Costes/hora	Horas dedicadas	Coste
Tello Miñana	15 €/h	1500	22.500 €
Luis Sánchez	30€/h	100	3.000€
Total			25.500

Tabla 20: Costes de personal

A.2.2 Costes de materiales

Se incluyen los costes de los materiales que hemos utilizado para llevar a cabo el trabajo de investigación. Dichos costes se pueden descomponer en el ordenador empleado y el software utilizado.

Equipo	Precio (€)
Se ha utilizado un PC[34] con arranque dual: Windows 7 y Ubuntu 12.04.3 LTS GNU/Linux , con Linux 3.8.0-31-generic	1.200
Total	1.200

Tabla 21: Coste del equipo empleado en el proyecto

Software

Se procede a comentar todo el software necesario para el desarrollo del proyecto. Algunas licencias empleadas son gratuitas al tratarse de software libre y gratuito, y otras son de pago.

[34] Ordenador personal

Concepto	Unidades	Precio(€)	Coste
Windows 7 Professional	1	137	137
Microsoft Office	1	80	80
Ubuntu 12.04.3	1	0	0
Java EE 1.7	1	0	0
Eclipse IDE	1	0	0
Stanford parser 3.2.0	1	0	0
Detector de idiomas de Google	1	0	0
Twitter4j-4.0.2	1	0	0
Total			**217 €**

Tabla 22: Coste de software

A.2.3 Presupuesto total del Trabajo Fin de Grado

A continuación se detalla el presupuesto final del proyecto, teniendo en cuenta el total de horas dedicadas y un coste estimado de horas / persona:

Concepto	Importe (€)
Costes de personal	25.500
Costes de material	1.200
Coste de software	217
Coste total	26.917

Tabla 23: Presupuesto total del Trabajo Fin de Grado

El presupuesto total del Trabajo Fin de Grado asciende a la cantidad de **VEINTISÉIS MIL NOVECIENTOS DIECISIETE** euros.

B. Manual de instalación y uso

1. Instalar el entorno de desarrollo integrado Eclipse y la máquina virtual de Java (versión 1.7), en nuestro caso hemos usado Windows 7.

   http://www.eclipse.org/downloads/

   http://java.com/es/download/

2. Para configuración la variable de entorno para comunicar Java con el sistema operativo hay que los siguientes pasos: Inicio -> Equipo -> botón derecho -> propiedades -> configuración avanzada del sistema -> variables de entorno -> modificamos el PATH: C:\Program Files\Java\jdk1.7.0\bin
3. Leer detalladamente la sección 4.2.1 de este trabajo para registrar una app en Twitter.
4. Descargar el paquete de Stanford para el lenguaje Java, el Proyecto Language Detection de Google code, la librería de Java para el API de Twitter (Twitter4j):
   - http://nlp.stanford.edu/software/lex-parser.shtml#Download
   - https://code.google.com/p/language-detection/wiki/Downloads
   - http://twitter4j.org/en/index.html#download
5. Añadimos una serie de jars externos a nuestro Proyecto Eclipse con los siguientes pasos:

   Botón derecho -> propiedades -> seleccionamos "Java Build Path" -> "Add External JARs" y añadimos los siguientes:

   Para utilizar el parser de Stanford usando el modelo del idioma inglés:
   - stanford-parser.jar:
   - stanford-parser-3.2.0-models.jar:

   En la detección del idioma del tweet necesitamos:
   - langdetect.jar
   - jsonic-1.1.2.jar

   Utilizamos el Api de Java para documentos de Microsoft añadiendo:
   - poi-3.5-FINAL-20090928.jar

   Para utilizar el REST API y el Streaming API, necesitamos:
   - twitter4j-core-3.0.6-SNAPSHOT.jar
   - twitter4j-stream-3.0.6-SNAPSHOT.jar

6. Buscar cuentas de usuarios en Twitter y almacenarlas en el fichero "todas_cunetas.txt". Introducimos las primeras 15 cuentas, ejecutamos la clase "GenerarTweets.java" para obtener los últimos 14000 tweets para cada usuario y esperamos 15 minutos, ya que hemos alcanzado el límite del REST API. Posteriormente, hacemos lo mismo con las siguientes 20 cuentas y así hasta las 200 que se obtienen de esta forma.
7. Ejecutamos la clase "CuentasAleatorias.java" para obtener 40 cuentas aleatorias utilizando el Streaming API. Luego ejecutamos otra vez "GenerarTweets.java" para obtener los tweets de estos usuarios nuevos (esperamos 15 minutos por cada 15 peticiones).
8. Instalamos en un sistema Linux el paquete R: sudo apt-get install r-base. Nosotros hemos utilizado Ubuntu 12.04.3 LTS GNU/Linux, con Linux 3.8.0-31-generic.
9. Damos permisos de ejecución para ejecutar el script de la siguiente forma: chmod +x script.r
10. Ejecutamos el script para generar los gráficos con los box-plots, ejecutar el script por cada identificador necesario. Ponemos el ejemplo para el identificador 1:
    - ./script.r fichero_ident1.txt ident1.eps
11. Abrir los gráficos con cualquier visor de postscript como Ghostview.